
150 ptas.
46
mi compuTer
CURSO PRACTICO DEL ORDENADOR PERSONAL,
EL MICRO Y EL MINIORDENADOR
AMSTRAD
64K COLOUR PERSONAL COMPUTER
CPC 464
CPC 464
Editorial Delta, S.A.

MI COMPUTER CURSO PRACTICO
DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona

Volumen IV - Fascículo 46

Director: José Mas Godayol
Director editorial: Gerardo Romero
Jefe de redacción: Pablo Parra
Coordinación editorial: Jaime Mardones, Francisco Martín
Asesor técnico: Ramón Cervelló

Redactores y colaboradores: G. Jefferson, R. Ford, F. Martín, S. Tarditti, A. Cuevas, F. Blasco
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Paseo de Gracia, 88, 5.°, 08008 Barcelona
Tels. (93) 215 10 32 / (93) 215 10 50 - Télex 97848 EDLTE

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador*, se publica en forma de 96 fascículos de aparición semanal, encuadernables en ocho volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

ISBN: 84-85822-83-8 (fascículo) 84-7598-005-8 (tomo 4)
84-85822-82-X (obra completa)
Depósito Legal: B. 52/1984

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 288411
Impreso en España - Printed in Spain - Noviembre 1984

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, 28034 Madrid.
Distribuye para Argentina: Viscontea Distribuidora, S.C.A., La Rioja 1134/56, Buenos Aires.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Edificio Bloque Dearmas, final Avda. San Martín con final Avda. La Paz, Caracas 1010.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (96 fascículos más las tapas, guardas y transferibles para la confección de los 8 volúmenes) son las siguientes:

a) Un pago único anticipado de 16 690 ptas. o bien 8 pagos trimestrales anticipados y consecutivos de 2 087 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 6.850.277 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Editorial Delta, S.A. (Paseo de Gracia, 88, 5.°, 08008 Barcelona), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Editorial Delta, S.A., en la forma establecida en el apartado b).

Para cualquier aclaración, telefonear al (93) 215 75 21.

No se efectúan envíos contra reembolso.





Mercancías internacionales

Steve Cross

Una atenta mirada al origen de los componentes de un ordenador puede revelarnos una sorprendente variedad de países

Fabricar micros es un negocio multinacional. El Amstrad CPC 464, por ejemplo, que describimos en la página 909, se construye enteramente en Corea y una gran proporción de los BBC Micro de Acorn se construyen en Hong Kong. Sinclair siempre ha seguido la política de ser sólo una empresa de investigación y diseño, subcontratando la fabricación de componentes y el ensamblaje final.

La razón de ello es la necesidad de construir las máquinas de la forma más barata posible. Las consideraciones sobre la posterior fabricación son de capital importancia para los diseñadores de ordenadores al principio de todo el proceso. Con el fin de mantener reducidos los costos, la placa de circuito impreso debe ser pequeña y simple. Esto significa que el diseño debe incorporar la menor cantidad posible de chips. La razón no es el costo de los chips en sí mismos, sino porque montar un gran número de éstos en una placa es caro y puede hacer que el producto final sea menos fiable.

Este último punto es el motivo por el cual en los micros más populares se utilizan chips ULA (*Uncommitted Logic Array*: disposición lógica no comprometida). El ULA, aunque es un chip caro de diseñar y construir, sustituye en la placa a docenas de chips más pequeños.

La mayoría de los microchips se fabrican en California, donde se ha acuñado el término Silicon Valley (Valle del Silicio) para referirse a la zona donde tienen sus sedes o sus centros de investigación la mayoría de las empresas norteamericanas dedicadas a la microelectrónica. Una vez fabricados estos chips, es necesario encastarlos dentro de la carcasa de plástico o cerámica. Esta parte del proceso no exige el mismo nivel de pericia técnica y es un trabajo de mano de obra intensiva. Puesto que ésta es más barata fuera de Estados Unidos, los chips se embarcan hacia diversos países.

Finalizado el diseño de la placa, comienza la búsqueda de un subcontratista. La fabricación de pla-

Crisol de razas
Este microordenador imaginario se fabricó con elementos procedentes de muchos países diferentes. Se fabrican chips en: 1) El Salvador; 2) Portugal; 3) Japón; 4) Estados Unidos; la carcasa, el teclado y el montaje final se efectúan en 5) Gran Bretaña; los conectores son de 6) República Federal de Alemania; el modulador de RF se produce en 7) Malasia, y la placa de circuito impreso (PCB) se fabrica en 8) Corea del Sur

cas de circuito impreso, al igual que la fabricación de chips, es una labor compleja que implica grandes inversiones en maquinaria. Existen muchas empresas que se especializan en la fabricación de dichas placas. A partir de copias heliográficas detalladas de la placa, el fabricante produce las familiares placas verdes con franjas metálicas para unir los componentes electrónicos.

Incluso el diseño de la propia placa está supeditado a las limitaciones del costo de fabricación. Es posible hacer construir placas de capas múltiples o multinivel, en las que se superponen varias capas de metal separadas por capas de aislante. Sin embargo, esto es caro y los diseñadores lo evitan. Para las placas de ordenadores siempre se especifica un sistema de patrón de agujeros, en el que los agujeros para todos los cables están metalizados para mejorar el contacto eléctrico, debido a la fiabilidad que proporciona este sistema al producto acabado.

Las fuentes de alimentación eléctrica, los moduladores de televisión, los conectores, teclados y otros componentes, se adquieren en distintos lugares de todas partes del mundo, siendo siempre los costos la principal consideración. Estos elementos se confían luego a otro subcontratista, a menudo extranjero, para el montaje final. Hasta la carcasa de plástico, que se fabrica con costosas maquinarias de moldeado, proviene de otro subcontratista.

El montaje final de un ordenador se puede realizar de dos maneras: ya sea por medios sumamente automatizados, o bien mediante una gran cantidad de mano de obra de reducido coste. En Estados Unidos, Europa y Japón generalmente se opta por la primera posibilidad, mientras que la segunda es común en Hong Kong, Singapur y Corea del Sur.

Las líneas de montaje automatizadas utilizan robots para instalar cada componente en las placas de circuito impreso. Los robots se alimentan con largas cintas de componentes, desde condensadores a chips de memoria. Lo que los operarios tienen que hacer es sustituir las cintas cuando éstas se vacían.

Sea cual fuere el sistema que se utilice, las placas son "atiborradas" con los chips adecuados y otros componentes, con sus cables sobresaliendo por debajo de la placa de circuito impreso. Las placas pasan luego a través de una máquina de "soldadura continua" que reviste con soldadura los cables sobresalientes. La soldadura se empuja hacia arriba a través de los agujeros metalizados de la placa y se solidifica para proporcionar una conexión fiable.

Luego las placas terminadas se verifican, se colocan en las carcasas, se empaquetan y se embarcan hacia los almacenes para su distribución, comercialización y venta a los clientes. Esto puede que parezca sencillo, pero cada una de las etapas del proceso posee sus propios problemas específicos.

El primer problema es de sincronización. Todos los componentes, procedentes de sus distintas fuen-

A través del globo

El subensamblaje, que vemos en la fotografía, es una floreciente industria en Gran Bretaña

Trabajadoras cualificadas utilizan microscopios para colocar circuitos en una PCB (Printed Circuit Board: placa de circuito impreso)



tes, deben llegar a un lugar listos para su ensamblaje. La persona responsable de esta operación, el comprador de componentes, es uno de los eslabones vitales de la cadena. Debe tener la habilidad para hacer un astuto trato comercial para adquirir los elementos de forma económica, así como para planificar y sincronizar las fechas de entrega. Todos los fabricantes de ordenadores son conscientes de los desastrosos efectos del retraso en la entrega de un solo componente. Las cadenas de producción sin utilizar cuestan dinero y los atrasos en las entregas significan pérdida de clientes.

El montaje del ordenador, ya sea automático o bien mediante mano de obra, es también un área susceptible de error. Es fácil insertar los componentes con la parte de arriba abajo o la de delante atrás, o, simplemente, omitirlos, arruinando por completo la placa final. La soldadura continua también se puede saltar una patilla de la carcasa de un chip. Del mismo modo, algunos de los componentes de una partida bien podrían no satisfacer sus especificaciones técnicas.

Estos problemas explican la necesidad de los controles, tanto de los componentes como de las placas acabadas. Muchos ensambladores de micros someten a pruebas selectivas los componentes que reciben, y todos ellos efectúan pruebas en las placas con diversos grados de sofisticación. La comprobación de placas es costosa, ya que requiere un poderoso hardware de ordenador. Pero es necesario realizar la inversión: un subcontratista no conseguirá que su contrato dure mucho si cuando entrega las máquinas éstas no funcionan.

El subcontratista que fabrica el Oric ha desarrollado un ingenioso control adicional. Las máquinas terminadas se pesan de forma individual. Si la máquina está por debajo del peso especificado, se deduce que durante el montaje se ha omitido algún componente. Éste es el motivo por el cual todas las cajas Oric poseen una etiqueta azul que indica el peso de la máquina.

La prueba final de un ordenador consiste en enchufar la máquina terminada en una fuente de alimentación eléctrica y un aparato de televisión. Los fabricantes de máquinas de oficina con frecuencia las dejan funcionando durante uno o dos días, "en remojo" o "al fuego". Esta prueba implica simplemente dejar la máquina ejecutando sus rutinas incorporadas o el software que viene con ella para asegurarse de que todo funcione adecuadamente.

Con toda esta cantidad de variables, es fácil apreciar por qué los micros personales pueden aparecer con retraso o no ser fiables. El montaje final depende de que los proveedores de chips y componentes hagan sus entregas a tiempo y según las especificaciones. Las empresas de diseño y marketing dependerán de que el ensamblador final entregue el producto acabado y sin tener los clientes esperando.



Reducción de costos
El objetivo del transporte de componentes para ordenador por todo el mundo es el de ahorrar capital al fabricante y no subir los precios de venta al público. Mediante la utilización de la mano de obra barata de otros países, los fabricantes han sido capaces de reducir significativamente sus costos. Pero recientes avances en la producción automatizada han hecho posible la producción de ordenadores completos en países más avanzados al mismo costo. El Oric Atmos, por ejemplo, se fabrica totalmente en Gran Bretaña, si bien Oric mantiene facilidades de producción en Singapur para sus mercados en el exterior

Componentes viajeros
Este mapamundi ilustra el movimiento de los componentes de microordenador durante el proceso de montaje

Chips fabricados en Silicon Valley (California)

CALIFORNIA

EL SALVADOR

Estas placas totalmente montadas aguardan la verificación final antes de ser instaladas

Kevin Jones

Movimientos claves

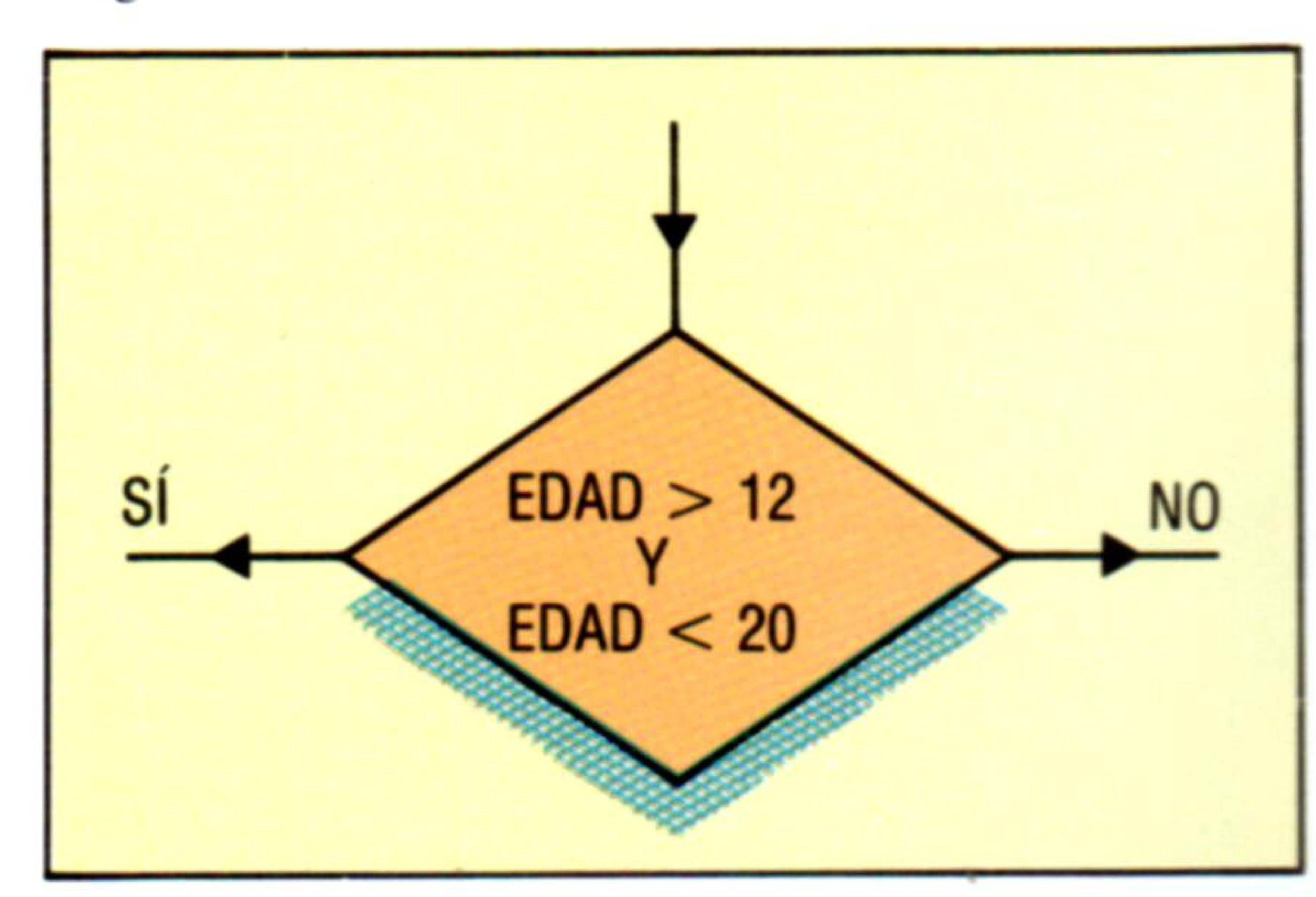
Continuamos con nuestro análisis de cómo utilizar eficazmente los diagramas de flujo en las etapas de planificación de programas

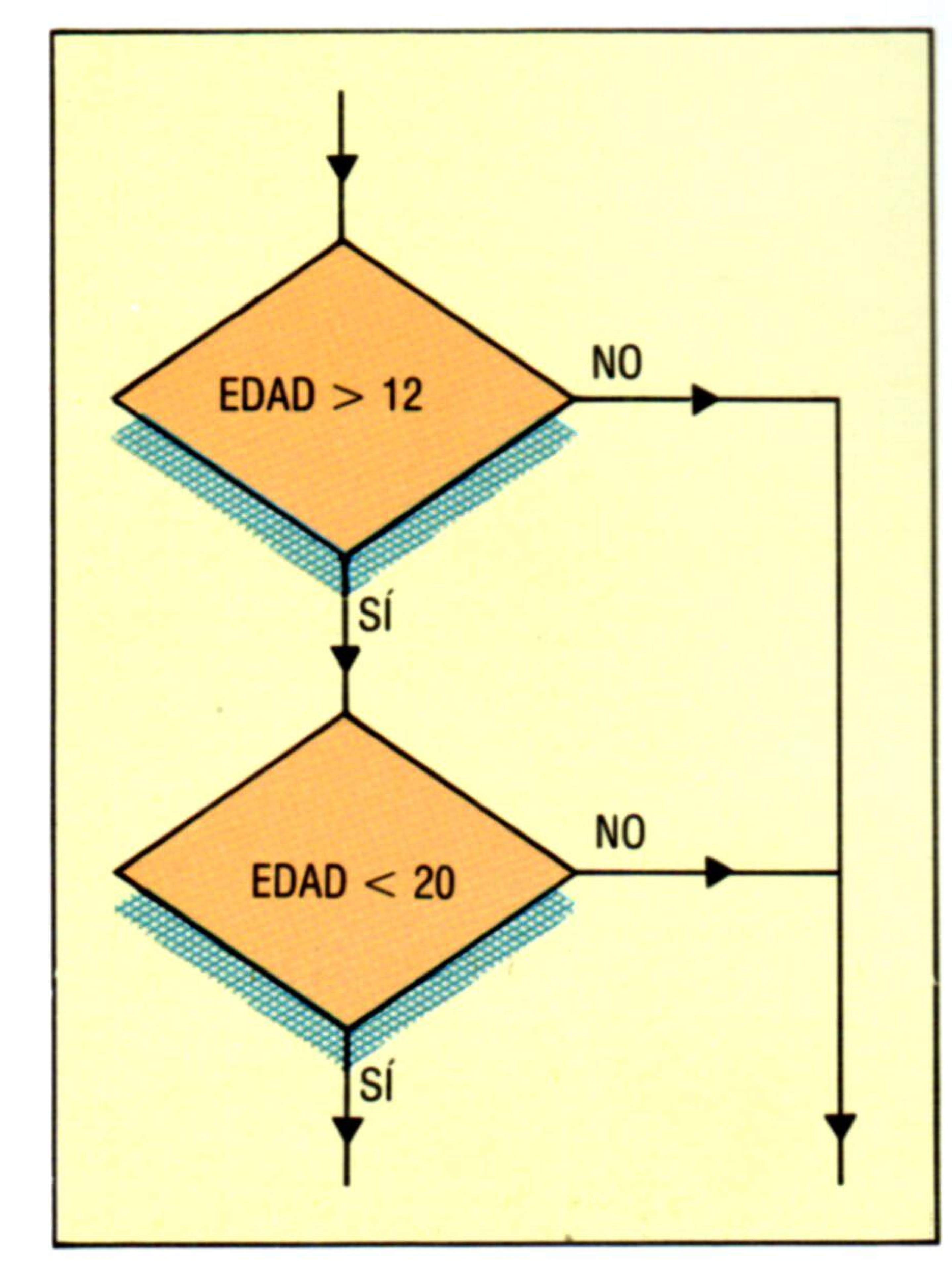
En el presente capítulo de técnicas de programación veremos cómo se pueden dividir las comparaciones compuestas en componentes simples y examinaremos la utilización de *tablas de decisión* en los casos más complejos.

Con frecuencia los programadores desean utilizar en sus programas comparaciones compuestas tales como:

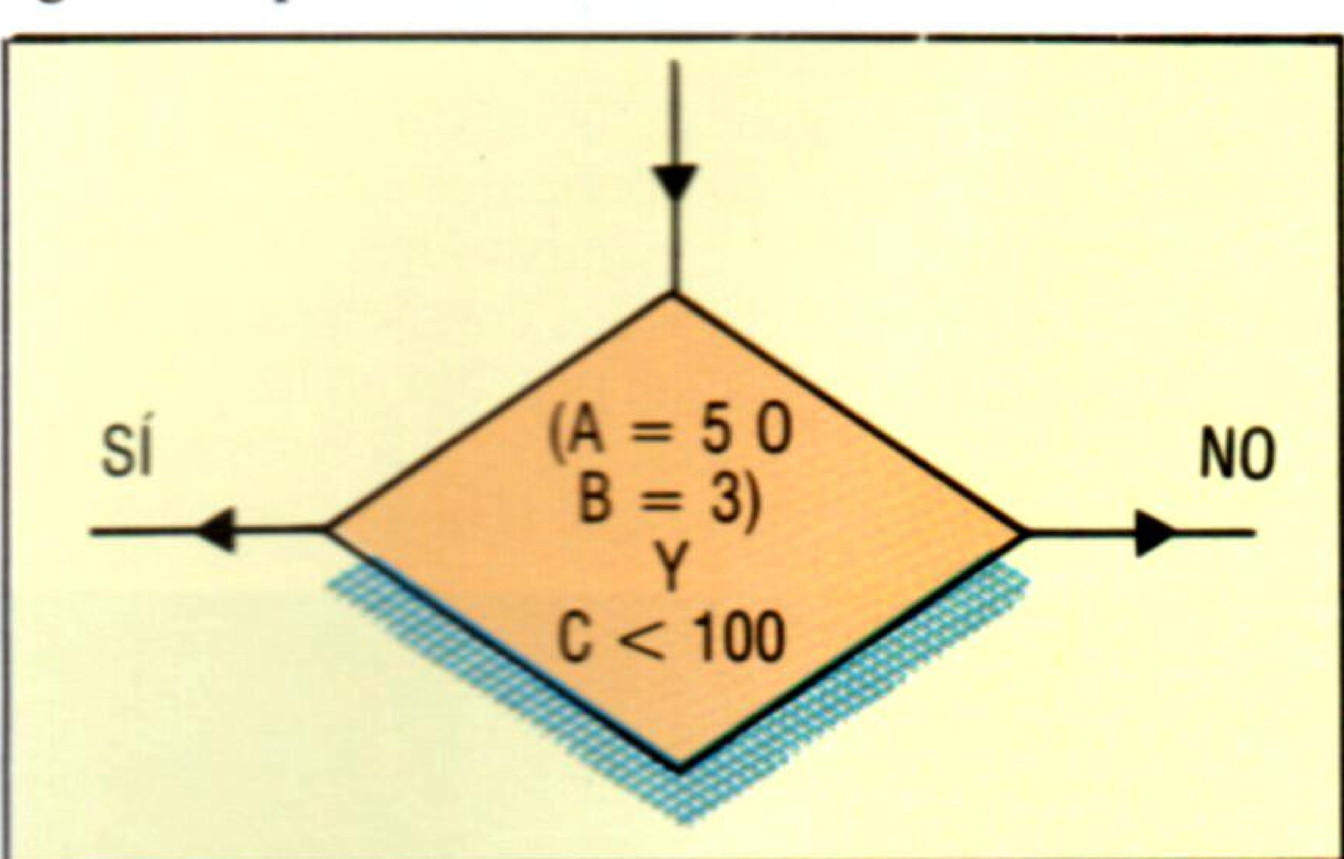
```
IF EDAD > 12 AND EDAD < 20 THEN ESTADO=
"ADOLESCENTE"
```

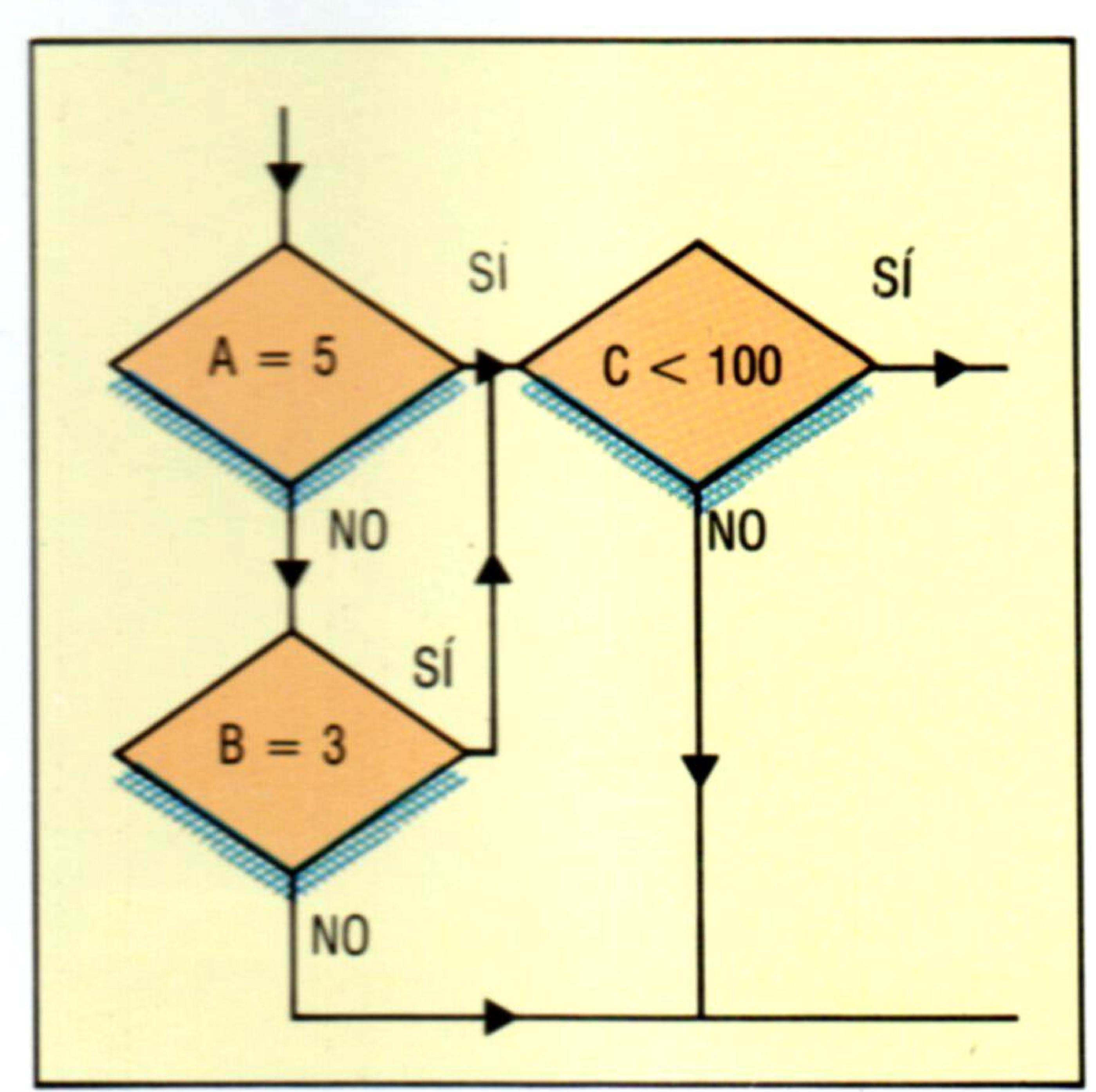
Los algoritmos que contienen instrucciones cómo ésta son más fáciles de entender si se divide la comparación compuesta en las comparaciones que la integran.





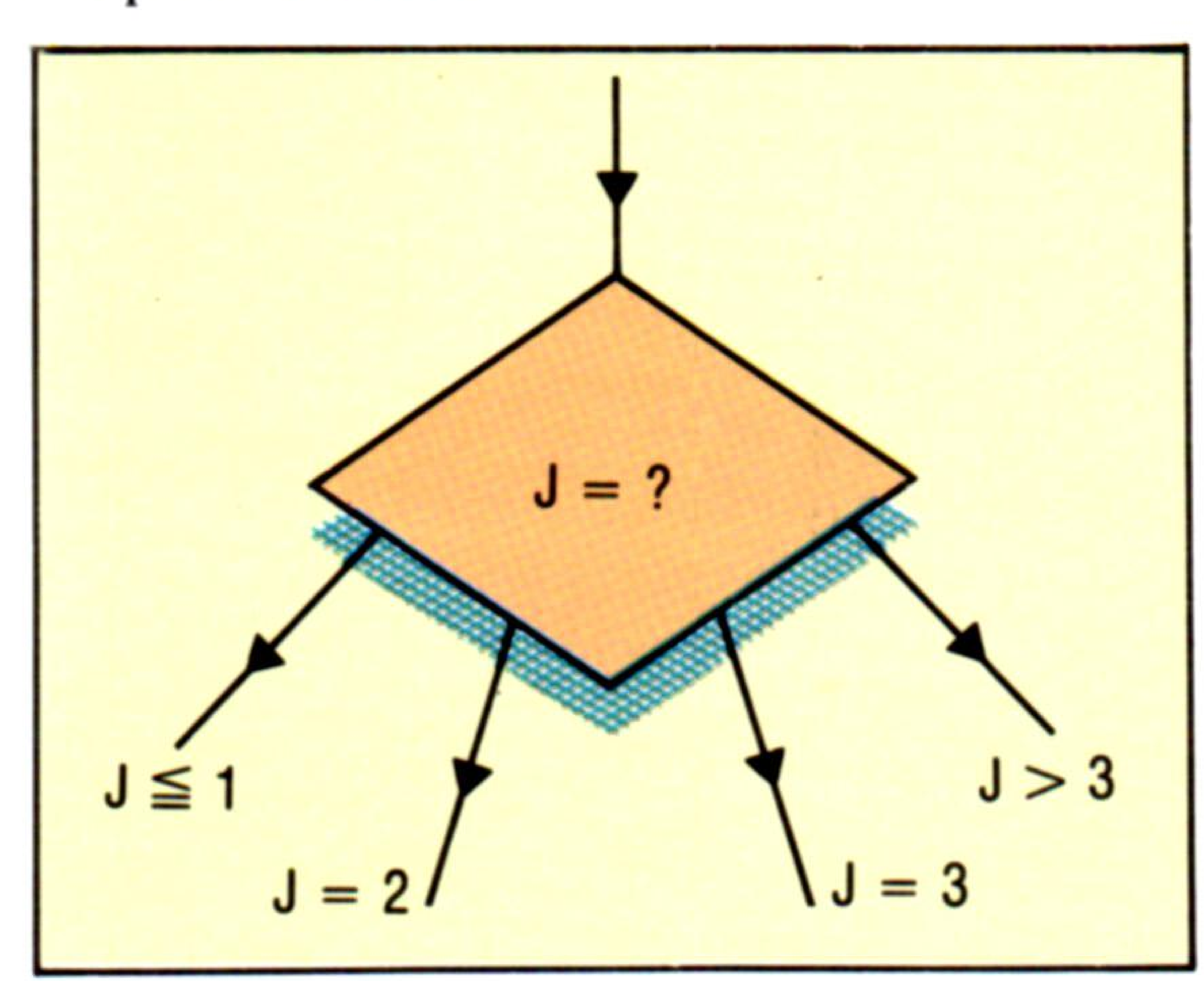
Este ejemplo en BASIC en forma diagramática ilustra cómo la versión compuesta es menos satisfactoria que la simple. El segundo ejemplo (abajo), con tres comparaciones simples, hace que el flujo de lógica a través de las casillas de decisión sea mucho más inteligible que su equivalente compuesto. También muestra una similitud con la lógica booleana, que permite la construcción de circuitos complejos a partir de una combinación de puertas lógicas simples.

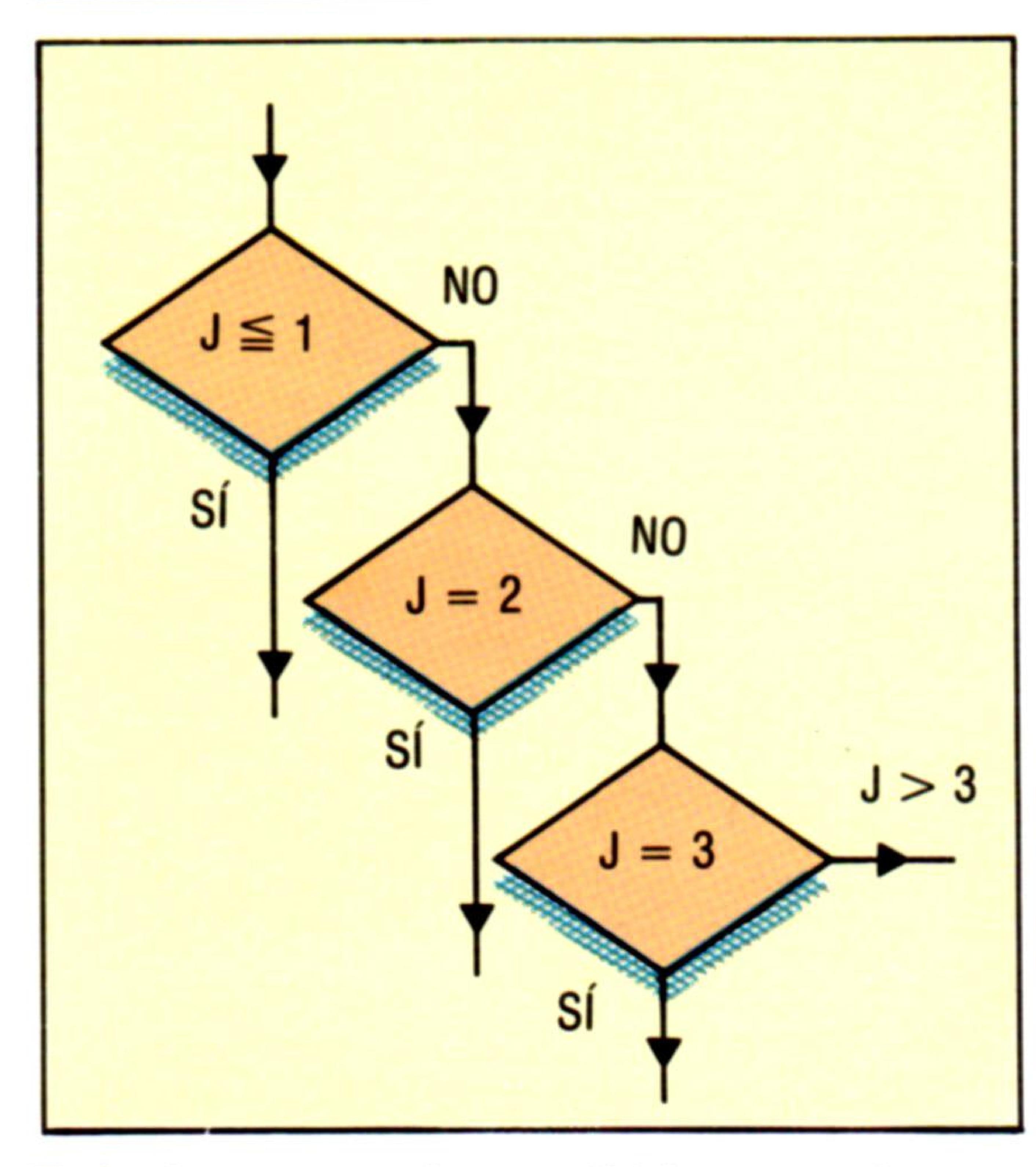




En los ejemplos, todas las comparaciones han sido binarias; no obstante, es bastante común que un algoritmo implique comparaciones con más de dos posibles salidas. Si tratáramos una entrada desde el teclado que representase la selección de una opción de un menú, entonces sería deseable bifurcar a una subrutina, de entre varias diferentes, para emprender la acción requerida. Para hacerlo, la mayoría de los lenguajes disponen de construcciones de bifurcación múltiple tales como CASE...OF en PASCAL y ON...GOTO y ON...GOSUB en BASIC. Las reglas para las comparaciones binarias también rigen para las múltiples: desde una casilla de decisión sólo se puede tomar una ruta y todas las salidas deben

estar bien etiquetadas, siendo todas las posibles rutas mutuamente exclusivas y cubriendo todas las posibilidades. Una comparación múltiple se podría dibujar, como en el ejemplo, con un conjunto de caminos de salida partiendo de la misma casilla de decisión. Sin embargo, es raro, y con mayor frecuencia la comparación será binaria, o sea, con sólo dos posibles salidas.





Todas las comparaciones múltiples se pueden representar como un conjunto de comparaciones binarias.

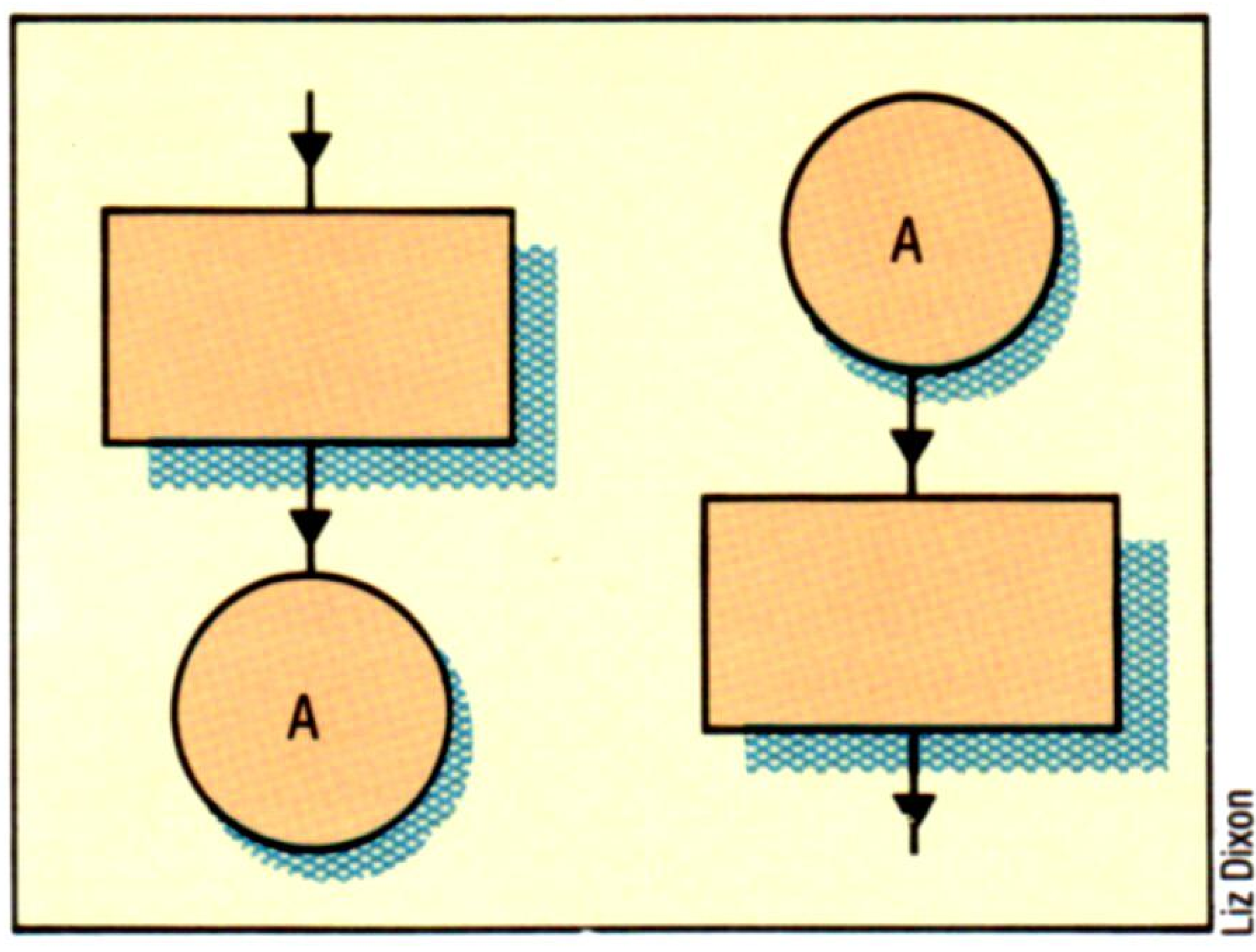
Como una alternativa a los diagramas de flujo, en especial cuando haya muchas comparaciones múltiples, podemos recomendar la utilización de tablas de decisión. Aquí ofrecemos un ejemplo de una tabla de este tipo, que representa un conjunto de reglas para realizar comparaciones. La tabla tiene cuatro secciones principales: texto que describe las condiciones para las reglas, texto que describe las acciones a emprender, una cuadrícula que muestra cómo las condiciones se adaptan a las reglas, y una cuadrícula que indica qué acciones son las apropiadas para cada regla. En la cuadrícula "condiciones/reglas", en los casilleros aparecen valores de variables, mientras que en la cuadrícula de "acciones/reglas" de abajo, un trazo indica qué acción se debe realizar y un valor actúa como un parámetro de entrada para dicha acción. La regla 4, por ejemplo, se lee: "Si se pulsa el botón de disparo y el nivel del juego es 2 y el nivel del jugador es novato, entonces activar escudo de explosiones aleatorias y reducir el nivel de energía en un 2 %".

	REGLAS							
CONDICIONES	1	2	3	4	5	6	7	8
BOTÓN DISPARO PULSADO	✓	✗	✗	✗	✓	✗	✓	✗
NIVEL JUEGO	1	1	2	2	1	1	2	2
NIVEL JUGADOR	NOVATO	NOVATO	NOVATO	NOVATO	EXPERTO	EXPERTO	EXPERTO	EXPERTO
ACCIONES								
ACCIÓN EXTRATERRESTRES							✓	✓
ESCUDO EXPLOSIONES			✓	✓			✓	✓
REDUCIR NIVEL ENERGÍA EN	1 %	1 %	2 %	2 %	2 %	2 %	4 %	4 %

Las tablas de decisión también sirven para combinar comparaciones simples y compuestas y, en formas más sencillas que la que hemos visto aquí, equivalen a las tablas de verdad que se emplean para predecir la salida de puertas lógicas.

Por último, he aquí una importante observación acerca de la utilización de los diagramas de flujo: siempre que sea posible limite la extensión de éstos a una sola página. Puede ser muy incómodo y ocupar mucho tiempo ir pasando una tras otra muchas hojas de papel. Si su algoritmo es demasiado extenso, intente dividirlo en otros más breves. Recuerde que cada algoritmo se puede utilizar como una única instrucción en algún otro algoritmo. De este modo, cada rutina de un programa se podría escribir como una única casilla de proceso en un diagrama de flujo de todo el programa, incluso si la rutina empleara otras rutinas que a su vez utilizaran otras, y así sucesivamente.

Es inevitable que algo vaya mal de vez en cuando y surja la necesidad de que un diagrama de flujo continúe en más de una página. Si esto sucede, corte el diagrama en un lugar apropiado (una comparación, p. ej.) y utilice un círculo con un símbolo identificador dentro para señalar el lugar donde el flujo de control continúa en la página siguiente (representado por otro círculo con el mismo símbolo en su interior, como vemos abajo). Si el control retorna al programa principal, vuelva a emplear círculos para señalar hacia atrás. Otra solución es contemplar la porción que falta como un algoritmo separado, referirse al mismo en una casilla de proceso y representarlo con su propio diagrama de flujo.



Haciendo conexiones

El modem Prism VTX5000 es uno de los accesorios de comunicaciones más ingeniosos en el mundo de la informática personal



La base de datos Prestel luchó por encontrar usuarios desde el día en que se puso, por primera vez, "en línea". Los equipos Prestel eran caros y la cantidad de información disponible en la base de datos era muy limitada como para ofrecer grandes ventajas en comparación con la que podía proporcionar un periódico. No obstante, con un modem y software adecuado, la mayoría de los micros personales pueden acceder al Prestel. Y así, con el objetivo de explotar esta posibilidad, Prism creó Micronet, un área separada dentro del Prestel que se dedica a noticias e información variada.

Pronto los modems y el software que posibilitaban el acceso del público al Micronet y al Prestel se pudieron conseguir a través de Prism, y el proyecto alcanzó un enorme éxito. La cantidad de usuarios y la gama de los servicios de información disponibles continúan creciendo.

En la producción de un accesorio para conectar el Sinclair Spectrum al Micronet, Prism se vio frente a un inmenso desafío, dado que la máquina no es adecuada para esta aplicación. No posee ni interface RS232 ni interface en serie, lo que significa que no se pueden conectar los modems normales. Posee una visualización en pantalla de 32 columnas por 24 líneas, y el Prestel exige una visualización de 40 por 24, así como el complicado sistema de gráficos de "teletexto". La empresa produjo una unidad "todo en uno" diseñada específicamente para esta única tarea: el modem Prism VTX5000.

La unidad contiene todas las interfaces necesarias para la conexión con el Spectrum, un modem de 1 200/75 baudios de conexión directa y el software para acceder al Prestel. No sólo proporciona las funciones estándar para conectarse (*log*) en el sistema, sino que también utiliza la pantalla para gráficos del Spectrum para imitar una auténtica visualización de teletexto de 40 por 24.

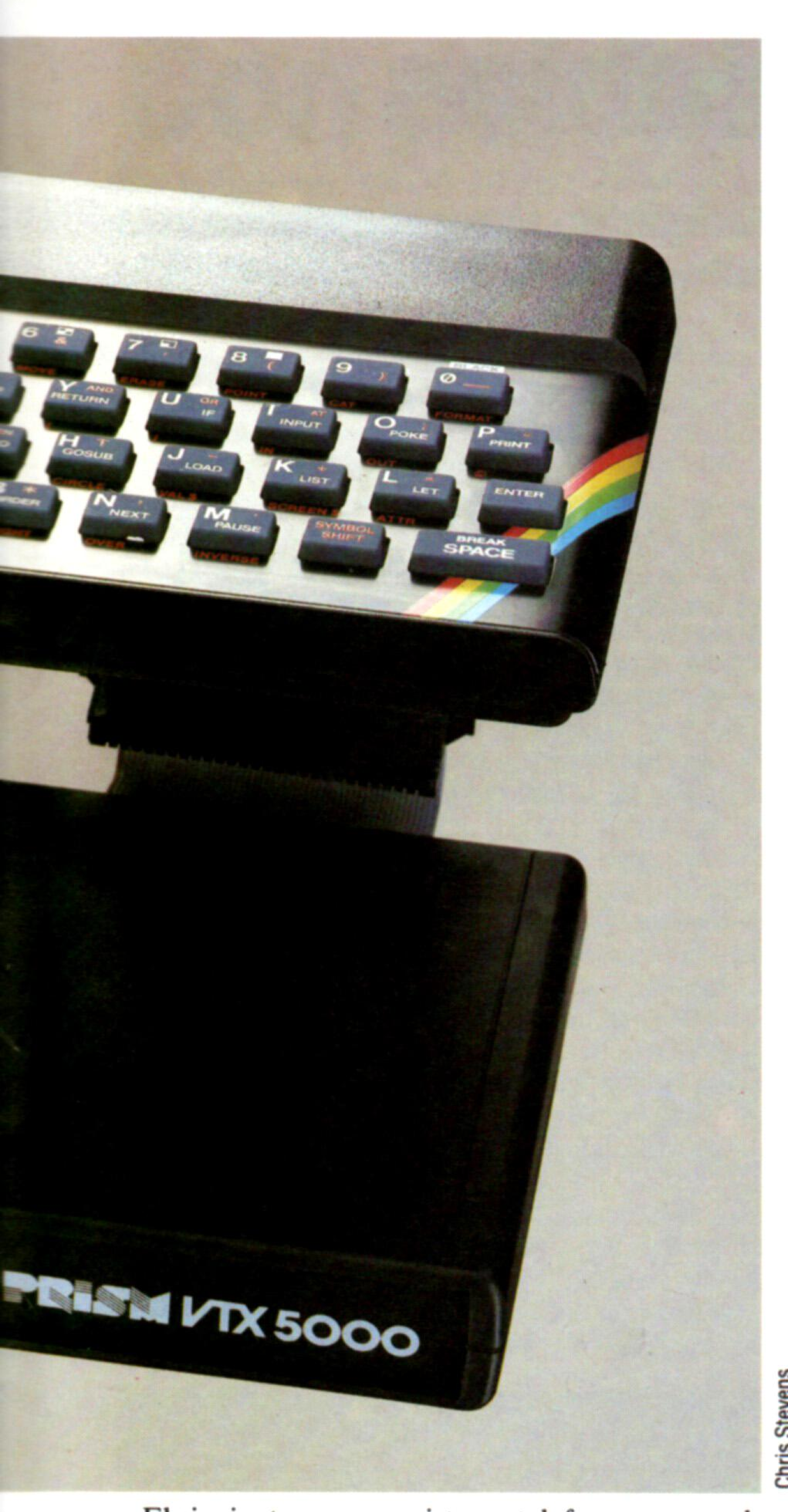
El VTX5000 se coloca debajo del Spectrum y se conecta a su conector de ampliación. El cable plano entre ambos posee un tercer conector, de modo que se pueden conectar otros periféricos del Spectrum, tales como una impresora o microdrives. Esta unidad se enchufa directamente en la red telefónica en vez de tener que utilizar un acoplador acústico (en el que el tubo del teléfono se encaja en dos tazas plásticas del modem). Esto proporciona una comunicación mucho más fiable.

Para instalar la unidad hay que disponer de unos enchufes telefónicos especiales. En caso de tener en casa un enchufe normal se debe instalar un conector apropiado. Se desenchufa el teléfono, se enchufa el modem y después se enchufa el teléfono al modem. Este método evita la necesidad de tener en su pared un enchufe telefónico de dos sentidos. Una vez conectado, el teléfono continúa funcionando normalmente.

Cuando se conecta el Spectrum, éste ejecuta automáticamente el software Micronet, que está almacenado en ROM dentro del modem, de manera que no se necesita cargarlo antes de utilizarlo. El paquete Micronet es muy similar al que proporcionan otros micros, de modo que una vez que lo haya utilizado no tendrá ningún problema para emplearlo en una máquina diferente. El software se controla mediante una serie de menús y es fácil de aprender y utilizar.

La primera opción del menú es conectarse (*log-on*), que significa entrar el número de identificación de 10 dígitos que se le otorga a cada usuario del Prestel. El ordenador recordará este número mientras esté encendido, de modo que sólo hay que teclearlo una vez en cada sesión, aun cuando el usuario llame al Prestel en varias ocasiones.





Chris Stevens

El siguiente paso consiste en telefonear a uno de los cuatro ordenadores que mantienen la base de datos Prestel. Cuando éste responde con una nota de tono agudo, se pulsa el interruptor Line (línea) del modem y ya puede colgar el teléfono. Ahora el Spectrum está "hablando" con el Prestel. Lo primero que se debe hacer es entrar una contraseña o palabra clave de cuatro letras. Este sistema impide que cualquier otra persona pueda utilizar la cuenta que usted tiene con el Prestel, y puede cambiarla cuantas veces desee para mantenerla en secreto.

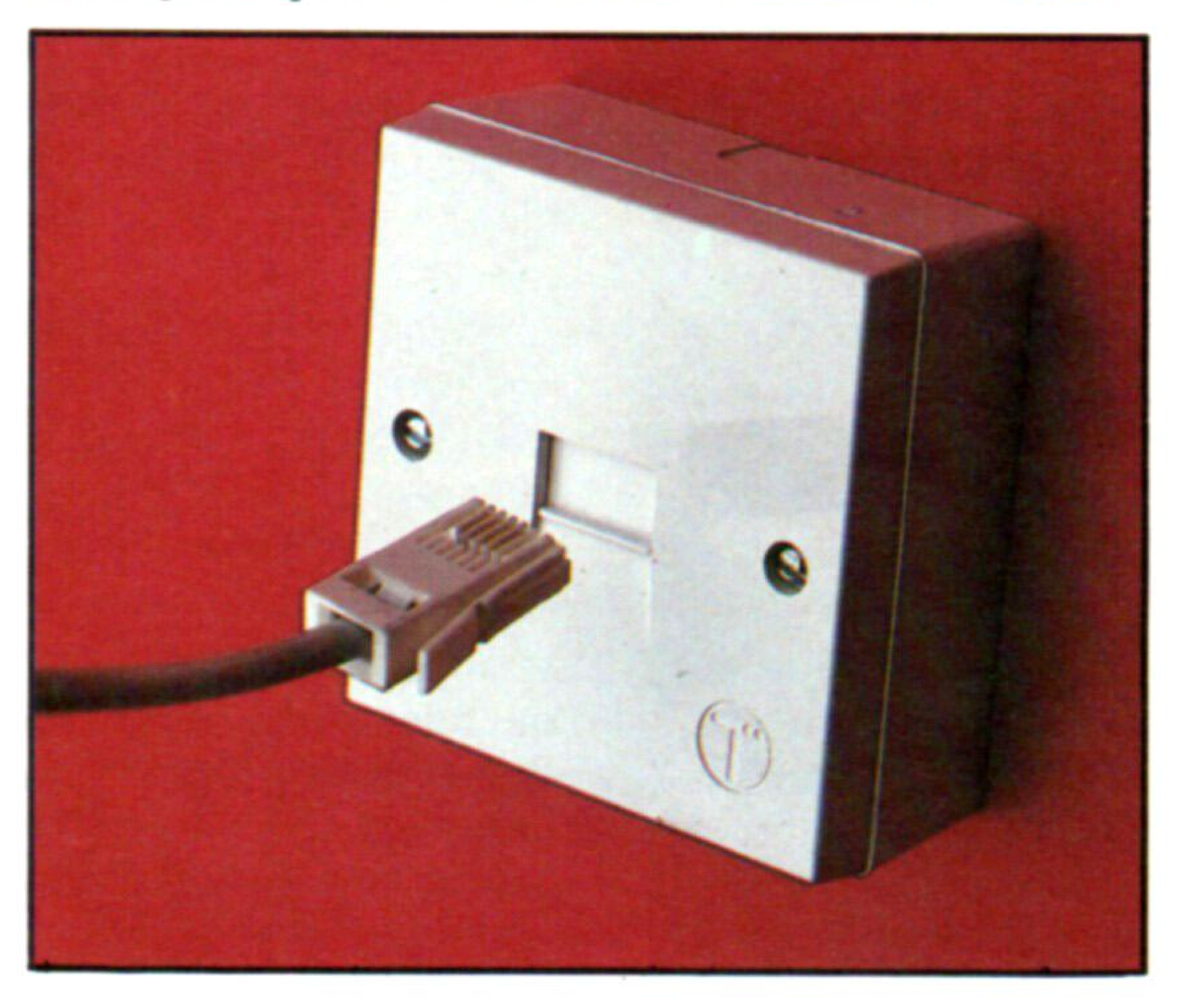
Una vez que se accede a la base de datos, es como utilizar cualquier otro adaptador Prestel. El Prestel requiere teclas de asterisco (*) y numérica (#) para ir pasando página a página. Para las mismas funciones, el Spectrum utiliza Enter y Symbol Shift. Se pueden llamar otras funciones del software Micronet en cualquier momento, siempre que siga conectado al sistema. Éstas permiten copiar páginas de la base de datos y almacenarlas en cinta o imprimirlas por impresora. Por otra parte, se pueden copiar programas completos del sistema en el Spectrum. Las páginas de telesoftware del Micronet incluyen programas gratuitos y programas por los que hay que pagar. Los gratuitos están bien para pasar el rato; pero, como es sabido, no se puede esperar gran cosa de lo que se ofrece gratis.

Tanto el Micronet como el Prestel ofrecen una variada gama de información. Además de noticias y reseñas, hay páginas de consejos técnicos, chistes, juegos, cartas, anuncios de contactos, etc. Incluso se puede enviar correo electrónico a otras personas que utilicen regularmente el servicio. El Micronet también posee un rival: el periódico *Viewfax*, en una parte distinta del Prestel, posee una irreverente sección de micros que es controlada por el misterioso "MicroGnomo".

La "explosión" informativa del Prestel viene muy bien como presagio de lo que será el futuro. Pero algunos de sus problemas originales siguen siendo evidentes, en especial en el Micronet. Muchas páginas de noticias ya son obsoletas cuando se incorporan a la base de datos y a menudo el camino que lleva de una página de información a la siguiente es confuso. Pero si usted se cansa del Prestel, puede tratar de comunicarse directamente con otros usuarios de Spectrum. El VTX5000 también está diseñado para enlazar dos ordenadores Spectrum a través de las líneas telefónicas, de modo que puedan enviarse mensajes y programas directamente el uno al otro. La velocidad de transmisión es de 1 200 baudios. Las primeras versiones del modem se vendían sin el software necesario para hacer esto, pero ahora se incluye con cada unidad una cinta que facilita esta transferencia de datos. Obviamente, este software sólo tiene algún valor para los usuarios de Spectrum que tengan amigos que también posean Spectrum y modems VTX5000.

Existen en uso otros varios estándares de comunicación para los modems, pero el VTX5000 no puede cumplir con todos ellos. La limitación más importante es que el modem carece del estándar necesario para los muchos tablones de comunicaciones y boletines gratuitos que están creando por todo el mundo los entusiastas de la comunicación por ordenador. Éstos operan a una velocidad de datos de 300 baudios, pero el VTX5000 no puede transmitir a esta velocidad tan lenta.

Quien realmente desee explorar toda el área de las comunicaciones por ordenador con su Spectrum quizá encuentre más adecuado comprar una interface RS232 (en especial la ZX Interface 1 de Sinclair) y emplear un modem de propósito general. Sin embargo, esto implicará la escritura de su propio software, conectar cables, etc. El VTX5000 es ideal para quienes sólo deseen utilizar el Prestel.



Conexión directa
El VTX5000 se conecta directamente al Spectrum sin necesidad de ninguna interface adicional. Su teléfono se enchufa en el VTX5000, que se engancha luego directamente en su conector telefónico modular. Observe que este modem sólo funciona en enchufes modulares y no en las cajas para conexión telefónica normales

Enchufes telefónicos modulares BT
El enchufe se introduce en una caja plástica cuadrangular, como se aprecia en la fotografía. Si usted no posee un enchufe modular, es necesario conseguir un enchufe de conversión

Valores variables

Un acumulador, a diferencia de un contador, que trabaja con valores constantes, puede verse aumentado o disminuido en cantidades variables

El diagrama del último programa "contador alfabético" mostrado en esta sección plasmaba una serie de resultados (total de consonantes, vocales, etc.) que habían sido obtenidos tras el incremento de unos contadores con unas cantidades constantes; así, cada vez que aparecía una "u", el contador correspondiente a las "úes" se incrementaba en uno hasta llegar a su valor final.

Pues bien, ésta es la función del contador, la de una variable (una posición de memoria) cuyo valor se incrementa o decrementa en función, siempre, de cantidades fijas: en el citado ejemplo, de 1 en 1, en el caso de los números pares de 2 en 2, o los años bisiestos, que lo hacen con un incremento de 4.

Si este dato fijo no fuera tal, es decir, si se empleasen cantidades diferentes entre sí para formar su valor, ya no se emplearía entonces el término de contador, sino que se daría paso al de *acumulador*.

Con lo cual puede establecerse que la diferencia principal y básica entre uno y otro es la de que mientras que el contador trabaja con valores fijos de incremento o decremento, el acumulador puede verse aumentado o disminuido en cantidades variables sin analogía entre ellas y que, aunque por casualidad, se repitan en alguna ocasión, ello no debe inducir a error, pues puede ser propiciado por pura coincidencia o en orden de alguna regla establecida de antemano que predisponga su esporádica repetición.

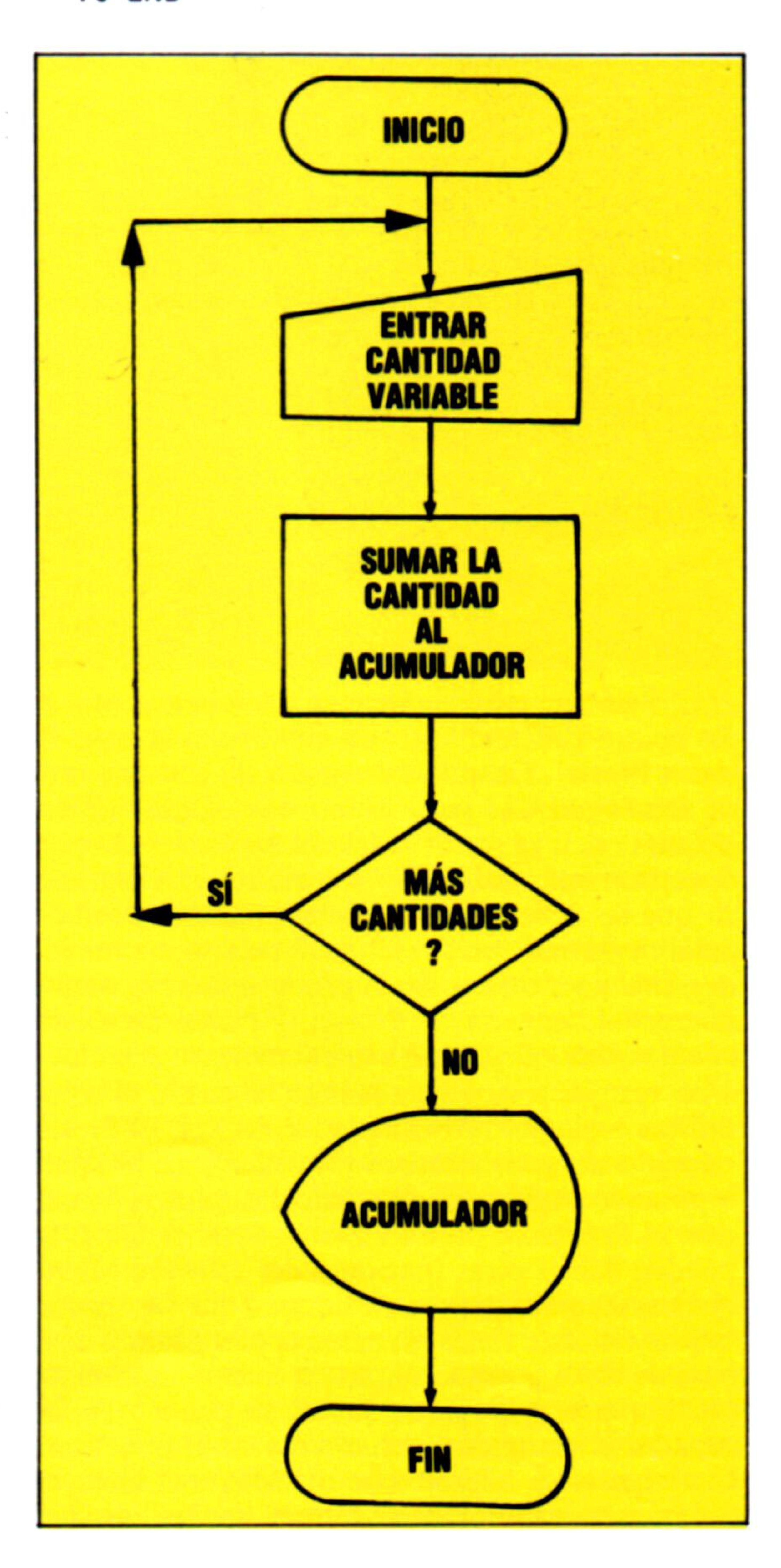
En el programa anteriormente citado, podrían haberse empleado diferentes acumuladores; así, por ejemplo, la función alfanumérica LEN(X$), que determina el número de caracteres de la serie X$, con lo aprendido hasta el momento estamos capacitados para suplirla, mediante el empleo de un contador inicial que irá incrementándose con el paso, uno a uno, de los diferentes caracteres. Pero existe otro procedimiento, propiciado por el uso de un acumulador que obtendría su valor final tras la suma de todos y cada uno de los diferentes contadores empleados.

Lo cual nos daría el total general obtenido con el resultado de sumar todos los totales parciales. En este caso los totales parciales representan datos variables y su contenido cambia según la frecuencia de aparición del carácter que cuenta.

Otro posible acumulador sería el que nos permitiera saber el número total de vocales aparecidas. Para ello, una vez acabado el recuento, se acumularía en una variable la suma conjunta de los cinco contadores.

Pero veamos ahora un pequeño ejemplo en el que se dispone de una serie de cantidades que, entradas por teclado una tras otra, van sumándose en un campo (AC). Al final del programa, es decir, una vez entradas todas las cantidades, se visualizará el contenido de dicho campo acumulador.

```
10 REM ACUMULADOR
20 INPUT"CANTIDAD";K
30 AC = AC + K
40 INPUT"HAY MAS CANTIDADES?(S/N)";F$
50 IF F$ = "S" THEN GOTO 20
60 PRINT "TOTAL ACUMULADOR:";AC
70 END
```





Con pantalla propia

Sus excelentes gráficos y su fiable hardware convierten al Amstrad en uno de los ordenadores más sofisticados del mercado

El ordenador Amstrad se vende en dos versiones. Ambos modelos son el mismo ordenador básico, pero se suministran con pantallas diferentes: una monocromática, la otra en color RGB. La fuente de alimentación eléctrica para el ordenador está alojada dentro de la carcasa de la pantalla y unos cables conectores llevan la energía y las señales de la pantalla hasta el ordenador. Como la grabadora de cassette también está incorporada en la carcasa del monitor, hay un único cable de energía eléctrica para abastecer a todo el sistema, de lo que resulta una mínima cantidad de cables de conexión.

La pantalla monocromática es de color verde y proporciona una visualización muy clara y bien definida, apta para tareas comerciales y otro tipo de tareas con textos. Se observa una ligera ondulación de la imagen, probablemente debida a la proximidad de la fuente de alimentación eléctrica en la carcasa de la pantalla. La pantalla en color sólo es de resolución media. Esto significa que, si bien puede visualizar en toda su plenitud los gráficos multicolores del Amstrad, no es capaz de visualizar texto de 80 columnas de una forma legible.

Para armonizar con la pantalla monocromática de estilo oficina, el Amstrad posee un teclado del tipo máquina de escribir completo, con un teclado numérico separado. Todas las teclas se pueden redefinir para crear caracteres diferentes a los que señalan las teclas. Además, el teclado numérico puede actuar como un conjunto de teclas de función programables. En cada una de éstas se puede programar una serie de hasta 32 caracteres. Por consiguiente se pueden utilizar teclas individuales para cargar o listar un programa o para borrar la pantalla.

Además de las características que el Amstrad tiene incorporadas, también hay prevista una amplia gama de periféricos. Una impresora Centronics se enchufa directamente en el micro a través de un conector marginal bastante tosco. También se pueden añadir unidades de disco, pero éstas todavía no han salido a la venta, si bien está anunciada para dentro de poco tiempo una unidad de disco que le proporcionará al ordenador memoria extra, el lenguaje LOGO y el sistema operativo de gestión CP/M. El Amstrad posee un único conector para palanca de mando que acepta las de tipo Atari. Algunos juegos exigen dos palancas de mando, de modo que Amstrad suministra un par que encajan simultáneamente en el conector.

El Amstrad tiene un altavoz incorporado (completo, con control de volumen), pero la señal de sonido también se puede enviar a un amplificador externo para obtener sonido estéreo a partir de las tres "voces" separadas del ordenador. Una voz se envía al altavoz derecho, una al izquierdo, y la tercera se mezcla a partes iguales entre ambos. Con



Chris Stevens

Opciones de pantalla
La versión más económica del Amstrad viene con una pantalla monocromática, pero una versión con pantalla en color tiene también un precio asequible. Las máquinas no se pueden adquirir sin las pantallas. Los dos programas que se están ejecutando en la fotografía son *Wordhang*, que es una versión de *Hangman* (Verdugo), y *Admiral Graf Spee*, un juego naval ambientado durante la segunda guerra mundial

una programación adecuada, una nave espacial extraterrestre no sólo se puede mover a través de la pantalla sino que da la impresión de que su sonido se desplaza por la habitación.

Controlar el sonido desde el BASIC es muy sencillo, a pesar de la sofisticación del sistema. No sólo se puede hacer que las notas empiecen y se interrumpan a una altura y volumen determinados, sino que también se puede controlar su envoltura o "forma". La envoltura de volumen se puede ajustar para hacer que una nota imite el sonido de, por ejemplo, un piano o una campana. La envoltura de tono también se puede controlar independientemente para la creación de efectos sonoros, como sirenas y silbidos, mientras cada voz se puede mezclar con "ruido", haciendo muy sencilla la simulación de explosiones y disparos de armas de fuego.

La mejor característica del Amstrad es la de sus excelentes gráficos. Hay tres modalidades de visualización disponibles, cada una de las cuales suministra una cantidad diferente de caracteres y de colores en pantalla al mismo tiempo. Cada una de estas modalidades utiliza los mismos 16 Kbytes de memoria y hay una compensación entre la cantidad de colores y la resolución y el formato del texto. La modalidad de resolución más alta permite sólo dos colores en la pantalla al mismo tiempo, uno de primer plano y otro de fondo. Hay disponible texto en 80 columnas y la máquina admite una resolución para gráficos realmente impresionante, de 640 × 200. En el otro extremo, la modalidad de 20 columnas admite 16 colores en la pantalla al mismo tiempo. La tercera modalidad admite 4 colores y 40 columnas.

Aunque la cantidad de colores que puede haber en pantalla al mismo tiempo es limitada, éstos se pueden seleccionar entre los 27 colores de la paleta. Se pueden visualizar marrones y tonos pastel además del rojo, azul, etc., habituales. El color del borde alrededor de la superficie de visualización se puede seleccionar de la misma paleta de colores. Se puede hacer que cualquiera de los 27 colores se enciendan de forma intermitente entre dos tonalidades diferentes a velocidades variables. Además de proporcionar un excelente marco de ampliación para imágenes detalladas, los gráficos del Amstrad también proporcionan una buena base para técnicas de animación. La única omisión la constituyen los sprites. No obstante, la manipulación de la pantalla mediante el BASIC es suficientemente rápida como para lograr que esta deficiencia no resulte tan importante.

A pesar de que la visualización en pantalla utiliza hasta 16 Kbytes de memoria, no roba espacio de memoria del programa del usuario. La RAM de la pantalla y la ROM de BASIC ocupan la misma área en el mapa de memoria del procesador. Un chip hecho a medida dentro del Amstrad conmuta entre ambas cuando ello es necesario, de modo que para los datos y los programas en BASIC quedan 42 Kbytes completos de memoria.

El BASIC Amstrad es una de las versiones del lenguaje más sofisticadas que existen. El excelente hardware para gráficos representa un gran apoyo. Hay varias características útiles para simplificar el trazado de imágenes en la pantalla. El origen de los gráficos se puede redefinir desde el rincón inferior izquierdo de la pantalla a cualquier punto dentro o

Interface para impresora Centronics
Es útil contar con esta interface para impresora estándar; lamentablemente Amstrad optó por utilizar un conector marginal para ella, en vez de un conector apropiado

Teclado numérico con teclas para el cursor

Puerta para palanca de mando

Chip de sonido GI

Conector del teclado

Chip de entrada/salida en paralelo 8225

Chip de ROM de 32 K
Contiene el sistema operativo de la máquina y la versión Amstrad del BASIC

Chip microprocesador Z80

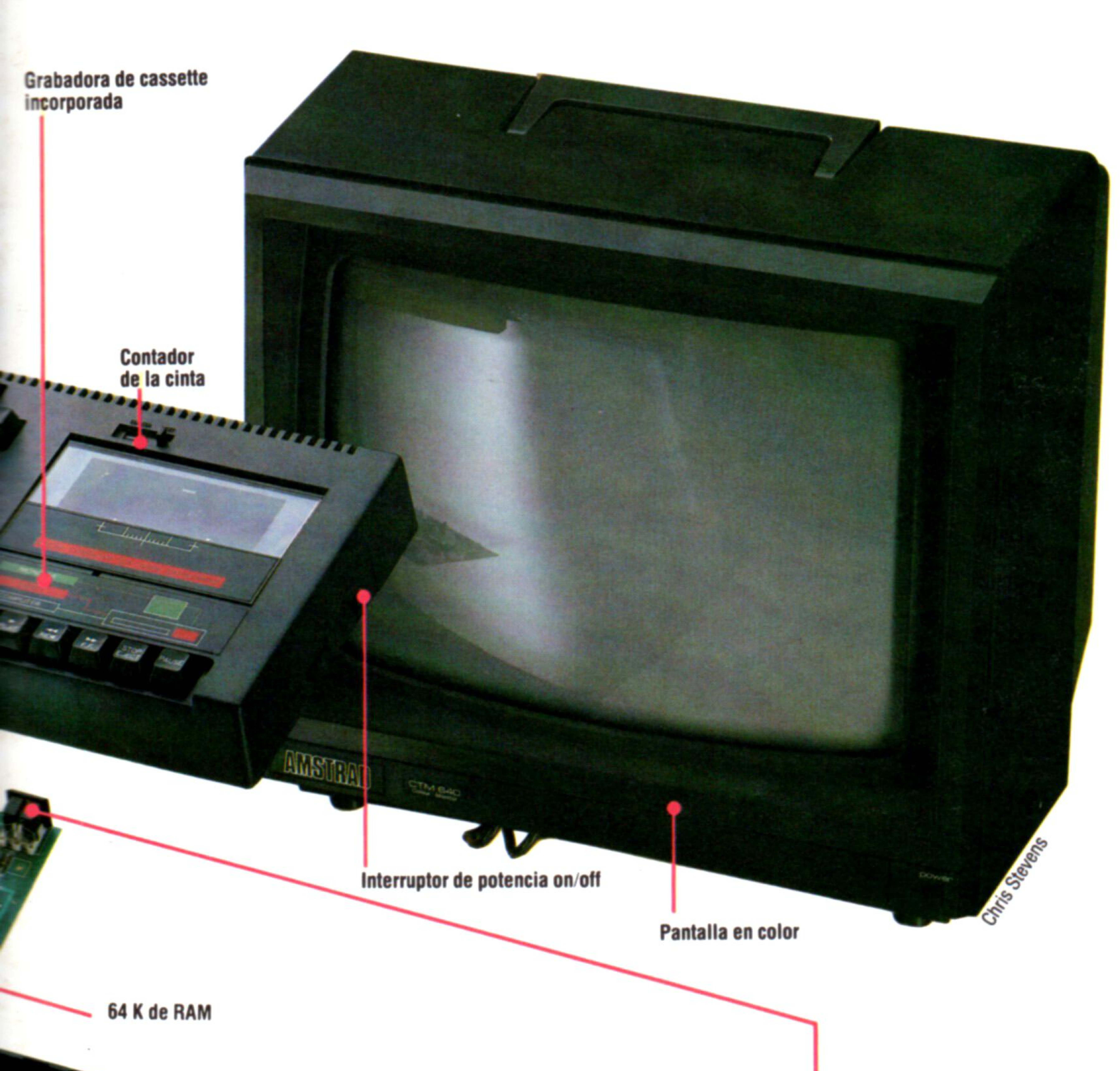
Chip de video 6845



Palancas originales

El Amstrad tiene un conector para palanca de mando que permite utilizar una del tipo Atari. No obstante, la firma fabricante también produce su propia palanca, y ésta tiene un conector que permite enchufar una segunda. Las señales provenientes de ésta se transmiten a través de la primera





Grabadora de cassette incorporada

Contador de la cinta

Interruptor de potencia on/off

Pantalla en color

64 K de RAM

Chris Stevens

Salida de pantalla
Este conector proporciona dos señales: una para una pantalla monocromática y otra para una pantalla en color RGB. Sea cual fuere el tipo de pantalla que se compre con el Amstrad, ésta se enchufa aquí

Puerta para ampliación
Especialmente útil para accesorios; el más destacado es la unidad de disco que permitirá al Amstrad convertirse en una máquina CP/M

AMSTRAD CPC 464

DIMENSIONES
Teclado-cassette:
565 × 170 × 70 mm
Pantalla en color:
380 × 350 × 350 mm

CPU
Z80

MEMORIA
64 K de RAM, de los cuales hay 42 K disponibles para programas en BASIC, 32 K de ROM

PANTALLA
Tres modalidades con mezcla total de texto y gráficos:
640 × 200 (2 colores)
320 × 200 (4 colores)
160 × 200 (16 colores)
Una paleta con 27 colores para escoger

INTERFACES
Palanca de mando (2), puerta para impresora Centronics, bus de ampliación (unidades de disco), salida de sonido en estéreo, salida para pantalla

LENGUAJES DISPONIBLES
BASIC (incluido), PASCAL (en cassette)

TECLADO
Tipo máquina de escribir, 74 teclas

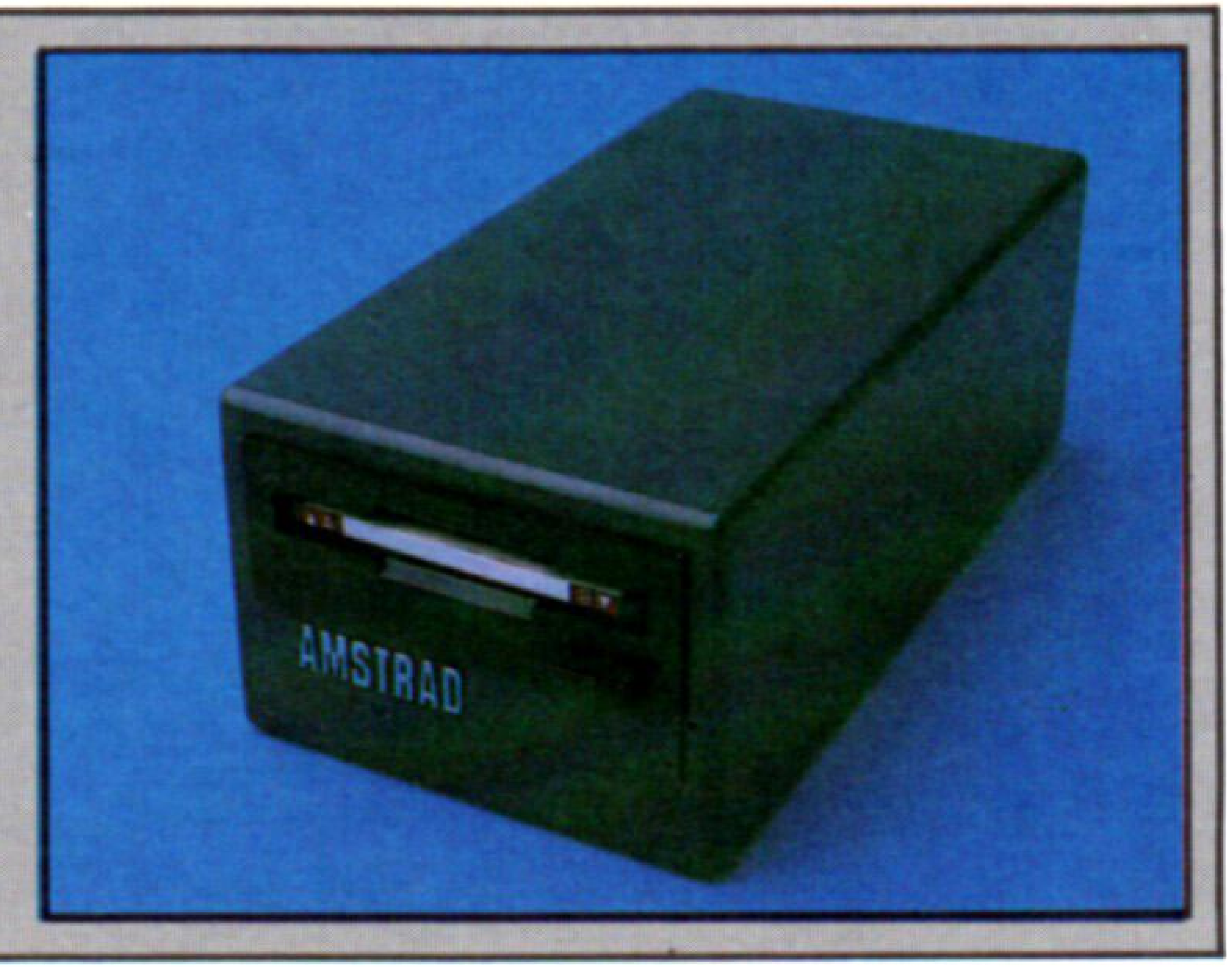
DOCUMENTACION
La guía para el principiante que se incluye con la máquina es fácil de entender. También se encuentra a la venta un manual de referencia de BASIC y un manual de referencias técnico

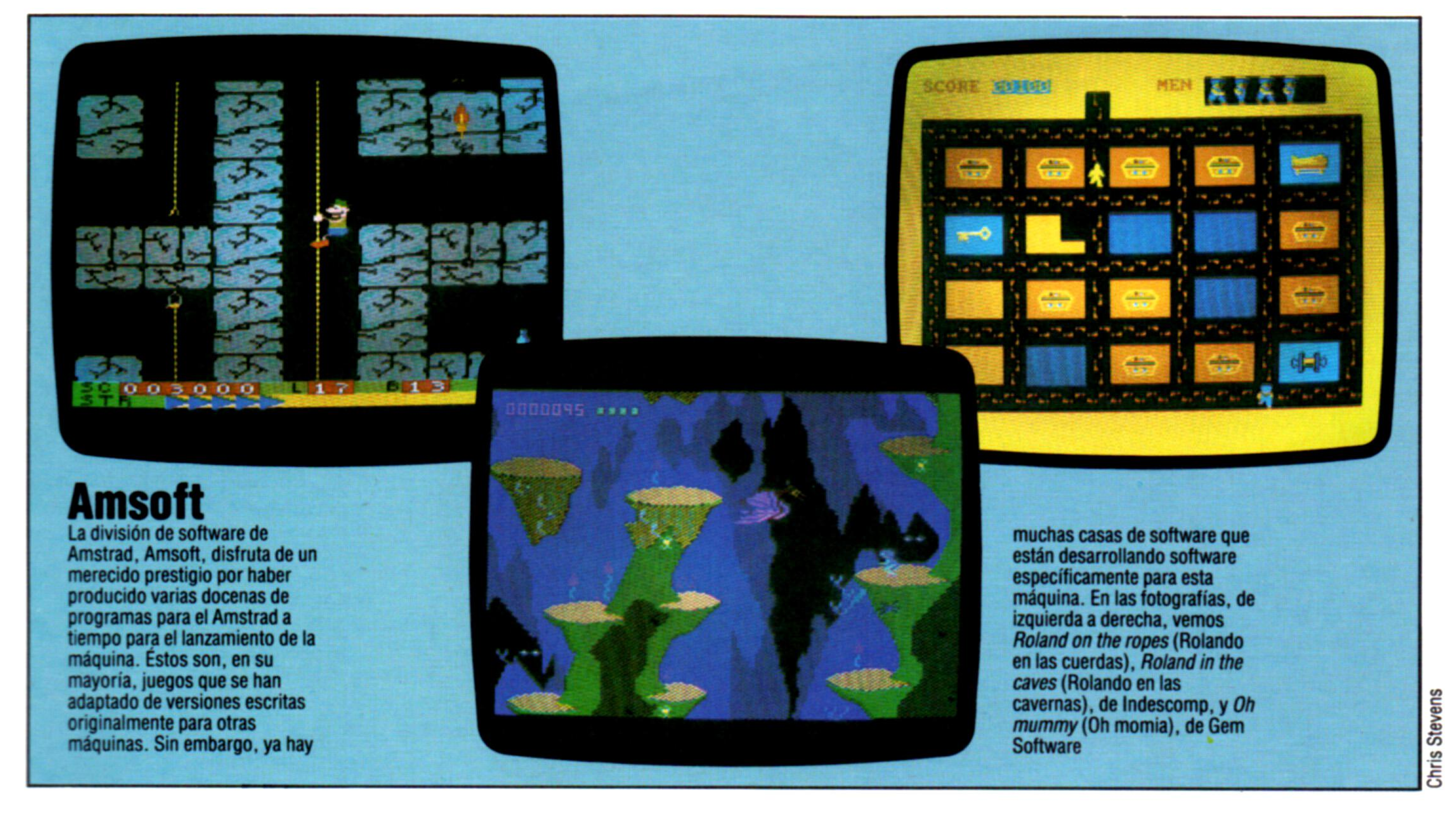
VENTAJAS
La pantalla y la grabadora de cassette incorporada hacen del Amstrad un sistema completo. Posee una gran memoria, gráficos versátiles y excelentes, y un sofisticado sonido estéreo

DESVENTAJAS
La grabadora de cassette no es de construcción muy sólida. Las instrucciones para gráficos del BASIC no están a la altura (todavía) del potencial que ofrece el hardware

Futura mejora

Amstrad tiene planeado producir un impacto en el extremo inferior del mercado de ordenadores de oficina lanzando una unidad de disco. Ésta incluirá el sistema operativo CP/M estándar, que necesitan la mayoría de los programas de gestión





Amsoft

La división de software de Amstrad, Amsoft, disfruta de un merecido prestigio por haber producido varias docenas de programas para el Amstrad a tiempo para el lanzamiento de la máquina. Éstos son, en su mayoría, juegos que se han adaptado de versiones escritas originalmente para otras máquinas. Sin embargo, ya hay muchas casas de software que están desarrollando software específicamente para esta máquina. En las fotografías, de izquierda a derecha, vemos *Roland on the ropes* (Rolando en las cuerdas), *Roland in the caves* (Rolando en las cavernas), de Indescomp, y *Oh mummy* (Oh momia), de Gem Software

Chris Stevens

fuera de la misma. Se puede establecer una ventana de gráficos para limitar las operaciones de gráficos a una pequeña porción de la pantalla. En la pantalla se pueden definir hasta ocho ventanas de texto al mismo tiempo, y el texto se puede dirigir a cualquiera de las ocho en cada momento. Se podrían enviar las preguntas a una ventana, por ejemplo, y las respuestas tecleadas a otra.

Los gráficos en BASIC también carecen de un procedimiento para rellenar con color una superficie de la pantalla. No hay instrucciones para dibujar zonas sólidas en la pantalla, ni tampoco para rellenar un esquema ya presente; sólo se pueden dibujar puntos y líneas. La única forma de producir un bloque de color es dibujar muchas líneas las unas muy cerca de las otras, método bastante ineficaz. No obstante, existe la posibilidad de que esta omisión se rectifique en un futuro cercano. Así como la ROM de BASIC se enciende y se apaga en el mapa de memoria para dejarle sitio a la memoria de la pantalla, se podría agregar otra ROM para que ocupara el mismo espacio. Las características de las que carece el BASIC Amstrad bien podrían aparecer en una ROM adicional y entonces las funciones nuevas se fundirían con BASIC antiguo. Lenguajes nuevos completos, como el PASCAL, FORTH y LOGO, se pueden añadir de la misma manera. Estas ROM "laterales" no ocuparían más espacio de memoria que el que ocupa el BASIC actual, de modo que también habría 42 Kbytes de RAM libres para su utilización. De hecho, se puede insertar memoria extra en el mapa de memoria de forma similar, de modo que se ampliarían los 64 Kbytes estándares de RAM.

El aspecto más original del BASIC Amstrad es su tratamiento de las "interrupciones". Muchos micros permiten que los programadores en lenguaje máquina utilicen el sistema de interrupciones incorporado en el sistema operativo de una máquina para activar sus propias rutinas en código máquina. El BASIC Amstrad lleva esta idea un paso más allá permitiendo utilizar las interrupciones desde el BASIC.

La instrucción del BASIC AFTER (después) cederá el control desde un programa a una subrutina especificada después de transcurrido un período de tiempo determinado. EVERY hará lo mismo de forma repetida. Esta avanzada característica hace que el escribir cualquier tipo de programa que dependa del tiempo, desde un programa para recogida de datos de laboratorio hasta un juego recreativo, sea más fácil y más eficaz.

El Amstrad es único entre los ordenadores personales por el hecho de que se suministra con una pantalla en vez de la visualización por el televisor que utilizan sus competidores. Las pantallas ofrecen mejor calidad de imagen, de modo que ésta es una ventaja muy importante. Sin embargo, es probable que los usuarios que hayan adquirido la versión monocromática necesiten contar ocasionalmente con una visualización en color. Tal como están las cosas, no existe ninguna forma de poder utilizar un televisor en color con el Amstrad, si bien hay a la venta, a modo de extra, un adaptador. Con el Amstrad se puede emplear una pantalla en color separada sin la ayuda del adaptador; pero, dado que el micro toma su energía a través de la pantalla monocromática, habría que tener las dos pantallas encendidas simultáneamente.

Con gráficos espléndidos y hardware fiable, un BASIC avanzado y potencial para ampliación, el Amstrad CPC 464 es uno de los ordenadores personales más sofisticados que existen actualmente en el mercado. También ofrece una excelente relación entre calidad y precio. Como inconveniente cabría destacar que la grabadora de cassette no es muy sólida y que las instrucciones para gráficos no están a la altura del hardware.

Selva y laberinto

"Sabre Wulf" tiene como protagonista a un aventurero, en busca de tesoros y en lucha constante contra sus atacantes

A causa de la caída en las ventas de los juegos de acción, las empresas de software han desviado su atención hacia una nueva forma de juego que combina elementos de la estrategia de los juegos recreativos con el sentido de la aventura. *Sabre Wulf*, el nuevo producto de Ultimate, Play The Game, responde a esta fórmula.

Un juego de aventuras, en el que el jugador debe conducir al héroe (a menudo un personaje extraído de la literatura de ciencia-ficción o de fantasía) a través de diversos lugares, resolviendo enigmas por el camino, con frecuencia tiene tramos laboriosos. La acción puede cesar durante prolongados períodos, mientras el jugador intenta hallar la respuesta de un enigma aparentemente insoluble. El juego recreativo, por otra parte, fomenta poco el pensamiento prolongado, exigiendo, en cambio, buenos reflejos y un dedo que dispare con rapidez.

Sabre Wulf intenta combinar lo mejor que poseen estos dos tipos de juegos. Básicamente se trata de un juego de laberinto que se desarrolla en un ambiente selvático, y sus orígenes guardan una estrecha relación con el clásico juego recreativo *Pac-Man*. El héroe, que recuerda a Indiana Jones (el aventurero protagonista de la película *En busca del Arca perdida*), es guiado a través de un laberinto sumamente complicado, evitando atacantes y apoderándose de tesoros para anotar puntos. El objetivo del ejercicio es recuperar los cuatro trozos desperdigados de un amuleto mágico roto.

El escenario de la jungla es magnífico, reproducido vívidamente en algunos de los gráficos más detallados que pueden verse en el Spectrum. Animales, plantas, montañas, cuevas y tesoros están todos ellos ilustrados de forma maravillosa y, en algunas ocasiones, el efecto global recuerda a un cuadro de Rousseau. El laberinto es extraordinariamente complejo y cubrir en una sola partida sólo una quinta parte del mismo constituye una proeza.

La mayoría de los repelentes seres atacantes se despachan simplemente con un rápido golpe de la espada del héroe, si bien él ha de estar justo enfrente de ellos en ese momento. En este sentido, *Sabre Wulf* imita a *Pac-Man*. Pero algunos contrincantes exigen armas especiales y éstas las debe ir descubriendo el jugador a medida que avanza por el laberinto. Otros objetos proporcionarán una vida adicional, lo cual es muy importante en este juego porque sólo los jugadores muy expertos tienen posibilidades de mantenerse ilesos durante mucho tiempo. Algunos objetos, como las orquídeas en flor, pueden ser tanto una ayuda como un obstáculo. Según el color que tengan, pueden hacerlo inmune al peligro, convertirlo temporalmente en un vegetal, doblar la velocidad a la cual se mueve el jugador o (lo que más lo confundirá) invertir el efecto de los mandos. Los efectos de las orquídeas desaparecen pronto.

Sabre Wulf está, en general, muy bien diseñado, aunque adolece de algunos de los defectos comunes que afectan también al software de otros juegos. El sonido, inicialmente muy atractivo y, por cierto, complejo, enseguida se vuelve molesto, y Ultimate no ha incluido la posibilidad de apagarlo. Si bien se supone que es un juego para uno o dos jugadores, en realidad es un juego para un solo jugador que posee dos marcadores. Está el usual Ultimate Hall of Fame (galería de famosos de Ultimate), con espacio para las iniciales de los seis marcadores más altos. Aquí Ultimate ha optado por el estilo recreativo para entrar los nombres: las iniciales se entran utilizando los controles de movimiento, ya sea en la palanca de mando o en el teclado.

Los fabricantes de software están empezando por fin a darse cuenta de que existe una amplia gama de teclados disponibles para el Spectrum y Ultimate ha previsto muchos de ellos. La utilización del teclado es menos satisfactoria porque *Sabre Wulf* emplea las teclas Q, W, E, R y T para movimiento y acción de la espada. Esto es difícil de comprender, puesto que todas estas teclas están en la misma fila y, por lo tanto, son extremadamente difíciles de utilizar.

Pero, dejando de lado errores soslayables acerca de la disposición del teclado, *Sabre Wulf* es un juego excelente que ofrece un notable equilibrio entre la acción de los juegos recreativos y la estrategia de los de aventuras.

Sabre Wulf: Para el Spectrum de 48 K
Editado por: Ashby Computers and Graphics
Autores: Ultimate, Play The Game
Palancas de mando: Kempston, Interface 2 y cursor
Formato: Cassette



La página del título
Esta pantalla muestra los gráficos de la página del título de *Sabre Wulf*. El mismo diseño está reproducido en la cubierta del paquete.
Sabre Wulf es un juego de laberinto con una ingeniosa variedad de personajes detestables. Los márgenes del laberinto están cubiertos por excelentes gráficos que imitan un paisaje selvático

Liz Heaney

Curso de colisión

Ahora examinaremos el control del movimiento y la detección de colisiones entre los caracteres del jugador y de las minas

El BASIC BBC posee no menos de cuatro instrucciones que responden a una sola pulsación de tecla. La elección de la instrucción dependerá, obviamente, del efecto deseado. INKEY$ e INKEY se utilizan normalmente cuando se desea esperar durante un período determinado una posible pulsación de tecla antes de seguir adelante con el resto del programa; GET$ y GET, por otra parte, siempre interrumpen la ejecución del programa hasta que se pulsa una tecla. Estas dos últimas instrucciones tienden a utilizarse cuando se requiere una respuesta a una pregunta, como "Otra partida S/N?". Al utilizar GET$ o GET, el programa aguardará hasta que se produzca una respuesta. En este caso, las únicas respuestas aceptables son "S" y "N". Podemos utilizar un bucle para repetir (REPEAT) la instrucción GET hasta (UNTIL) que la respuesta sea "S" o "N":

```
1000 PRINT "OTRA PARTIDA S/N?
1010 REPEAT
1020    A$ = GET$
1030 UNTIL A$ = "S" OR A$ = "N"
1040 Etc.
```

Si se utilizan GET$ o INKEY$, entonces la tecla pulsada se interpreta como un carácter alfanumérico (tipo string), como en el ejemplo anterior. Si utilizamos GET o INKEY, entonces se devuelve un carácter de tipo numérico; este valor es el código ASCII de la tecla pulsada. Estas opciones permiten que el programador pueda preguntar por la pulsación de aquellas teclas que no tienen un carácter asociado, como las teclas Return o las de movimiento del cursor. La sentencia *FX4,1 se podría usar para que las teclas del cursor devuelvan códigos ASCII. De hacerlo así, las teclas tendrían los valores:

Cursor izquierda	136
Cursor derecha	137
Cursor abajo	138
Cursor arriba	139

Supongamos que deseamos aceptar para nuestro programa pulsaciones de teclas de cursor izquierda y cursor derecha. El siguiente segmento de programa utiliza INKEY para esperar una entrada durante un cuarto de segundo:

```
1000 *FX4,1:REM ENCENDER MODALIDAD ASCII
     DEL CURSOR
1010 REPEAT
1020    A = INKEY(25)
1030 UNTIL A = 136 OR A = 137
1040 *FX4,0:REM RESTAURAR CURSOR A
     MODALIDAD EDICION
```

El parámetro 25 de la línea 1020 le dice al ordenador que espere 25 centésimas de segundo antes de seguir adelante con el programa.

Estas sentencias no leen del teclado mismo sino que lo hacen en un área de memoria dentro del ordenador llamada *buffer del teclado*. Éste es un espacio de almacenamiento temporal para caracteres que se introducen desde el teclado, y es algo así como la cola para entrar en un cine. Los nuevos caracteres tecleados se ponen al final de la cola y el procesador va tomando caracteres del comienzo de la misma. De esta forma, si se teclean caracteres más rápidamente de lo que puede manipularlos el procesador, éstos no se pierden sino que tan sólo aguardan su turno en el buffer del teclado. Puesto que INKEY, GET, INKEY$ y GET$ normalmente leen sobre la parte delantera de la cola del buffer del teclado, no hay forma de saber durante cuánto tiempo ha estado un determinado carácter esperando en el buffer para ser procesado. En los juegos que se controlan desde el teclado, ello puede significar una respuesta lenta, ya que el programa puede estar procesando anteriores pulsaciones de teclas mientras el jugador está realizando otras nuevas. Por ejemplo, si se llena el buffer del teclado con códigos de cursor derecha, será necesario procesarlos todos antes de que el programa pueda responder a una instrucción cursor izquierda. Esto puede dejar al jugador pulsando frenéticamente el botón cursor izquierda ¡y preguntándose por qué razón el objeto que controla se mueve hacia la derecha!

Existen dos soluciones para este problema. La primera consiste en limpiar siempre el buffer del teclado justo antes de investigarlo. Esto se puede efectuar utilizando la sentencia *FX15,1. Alternativamente podemos utilizar otra variación de INKEY. Como hemos descrito más arriba, INKEY() espera durante un período de tiempo, determinado por el número entre paréntesis, a que se pulse una tecla antes de continuar con el programa. Podemos, sin embargo, hacer que INKEY lea directamente del teclado en vez del buffer del teclado especificando entre los paréntesis que siguen a la instrucción un número *negativo*. Con esta finalidad, todas las teclas tienen asignado un número negativo. En nuestro programa emplearemos las teclas del cursor para controlar el movimiento. Los valores de las teclas a utilizar con INKEY son:

Cursor izquierda	−26
Cursor derecha	−122
Cursor abajo	−42
Cursor arriba	−58

El siguiente procedimiento utiliza INKEY para leer del teclado directamente para cada una de las cuatro teclas del cursor sucesivamente. Si se está pulsando una de las teclas, entonces se accede a otro procedimiento ("mover"), pasando dos parámetros. Estos parámetros contienen información relativa a la dirección en la cual se ha de desplazar el carácter que representa el detector de minas.



```
3000 DEF PROCleer—teclado
3010 REM ** ARRIBA? **
3020 IF INKEY(-58) = -1 THEN PROCmover(0,-1)
3030 REM ** ABAJO? **
3040 IF INKEY(-42) = -1 THEN PROCmover(0,1)
3050 REM ** DERECHA? **
3060 IF INKEY(-122) = -1 THEN PROCmover(1,0)
3070 REM ** IZQUIERDA? **
3080 IF INKEY(-26) = -1 THEN PROCmover(-1,0)
3090 ENDPROC
```

El procedimiento "mover"

Este procedimiento es el cuerpo del programa. Con él se desplazan los caracteres del detector y del ayudante y se verifican las colisiones con minas. Vamos a analizar en primer lugar la sección que controla el movimiento de los caracteres.

Desde el procedimiento "leer—teclado" se le pasan a "mover" dos parámetros. Éstos se aceptan en las variables delta-x y delta-y para su utilización en "mover", y corresponden a los cambios a efectuar en las coordenadas x e y del detector de minas. Por ejemplo, si se pulsara la tecla cursor arriba, entonces se le pasarían a "mover" los valores 0 y −1. Las instrucciones xdet = xdet + delta−x e ydet = ydet + delta−y harían que se actualizaran las coordenadas del detector. En el caso de cursor arriba, se le suma 0 a xdet y −1 a ydet, lo que en realidad es restarle uno a su valor. Esto parecería implicar el movimiento de una unidad hacia *abajo* en la pantalla, pero debemos recordar que el origen de las posiciones de los caracteres está en el rincón superior izquierdo y que los valores de y se incrementan hacia abajo en la pantalla. Por consiguiente, decrementar ydet en uno produce el movimiento hacia arriba de una celda de carácter. Tal vez usted quiera verificar que los valores pasados para las otras tres direcciones corresponden, realmente, a las alteraciones correctas de xdet e ydet. Sería factible utilizar este sistema para incluir movimientos en diagonal. Pasándole a "mover" los valores (1,−1) el detector se movería oblicuamente una celda hacia arriba y una hacia la derecha. Sin embargo, se habrían de introducir otras teclas para permitir el control en diagonal desde el teclado. Veamos el listado para el procedimiento "mover":

```
3220 DEF PROCmover(delta-x,delta-y)
3230 REM ** BORRAR POSICIONES VIEJAS **
3240 COLOUR 1
3250 PRINTTAB(xdet,ydet);"  "
3260 PRINTTAB(xhom,yhom);"  "
3270 REM ** MOVER DETECTOR **
3280 xdet = xdet + delta-x
3290 ydet = ydet + delta-y
3300 REM ** VERIFICAR LIMITES **
3310 IF xdet > 17 THEN xdet = 17
3320 IF ydet > 25 THEN ydet = 25
3330 IF xdet < 2 THEN xdet = 2
3340 IF ydet < 1 THEN ydet = 1
3350 REM ** CALCULAR COORDS. HOMBRE **
3360 xhom = 19-xdet
3370 yhom = 26-ydet
3380 PROCconvertir(xhom,yhom)
3390 IF POINT(xgraf,ygraf) = 2 THEN PROCexplotar(xgraf,ygraf)
3400 PROCconvertir(xdet,ydet)
3410 IF PONT(xgraf,ygraf) = 2 THEN PROCdescubierta-mina
3420 PROCsituar-sujetos
3430 ENDPROC
```

Antes de alterar las coordenadas x e y del detector debemos borrar las posiciones antiguas del detector y el ayudante. Las líneas 3250 y 3260 utilizan los valores antiguos de xdet, ydet, xhom e yhom para imprimir (PRINT) espacios sobre los caracteres viejos. Dado que los nuevos caracteres se imprimirán (PRINT) en rojo (color lógico 1), se utiliza la instrucción de color en la línea 3240 para establecer el color de fondo en curso en 1. Las líneas 3280 y 3290 actualizan las coordenadas del detector tal como hemos descrito anteriormente. Antes de imprimir (PRINT) realmente el detector en su nueva posición, se deben realizar comprobaciones para asegurarnos de que no estemos incrementando o disminuyendo coordenadas por fuera de la zona que hemos definido como nuestro campo de minas. Los límites superior e inferior de xdet e ydet se comprueban entre las líneas 3310 y 3340. Aquí se ha decidido que si el detector llega a un margen, entonces permanecerá allí hasta que se lo mueva en la dirección contraria. Por ejemplo, la línea 3310 verifica si se ha alcanzado el margen derecho del campo de minas, señalado con coordenada x de 17. Si se realizara un intento por incrementar xdet por encima de 17, entonces esta línea simplemente restauraría el valor a 17. Hubiera sido igualmente posible crear un efecto de "rodillo" según el cual el detector, al llegar al margen derecho, apareciese seguidamente por el margen izquierdo de la pantalla. Para producir este efecto en el margen derecho del campo de minas, alteramos la línea 3310:

```
3310 IF xdet > 17 THEN xdet = 2
```

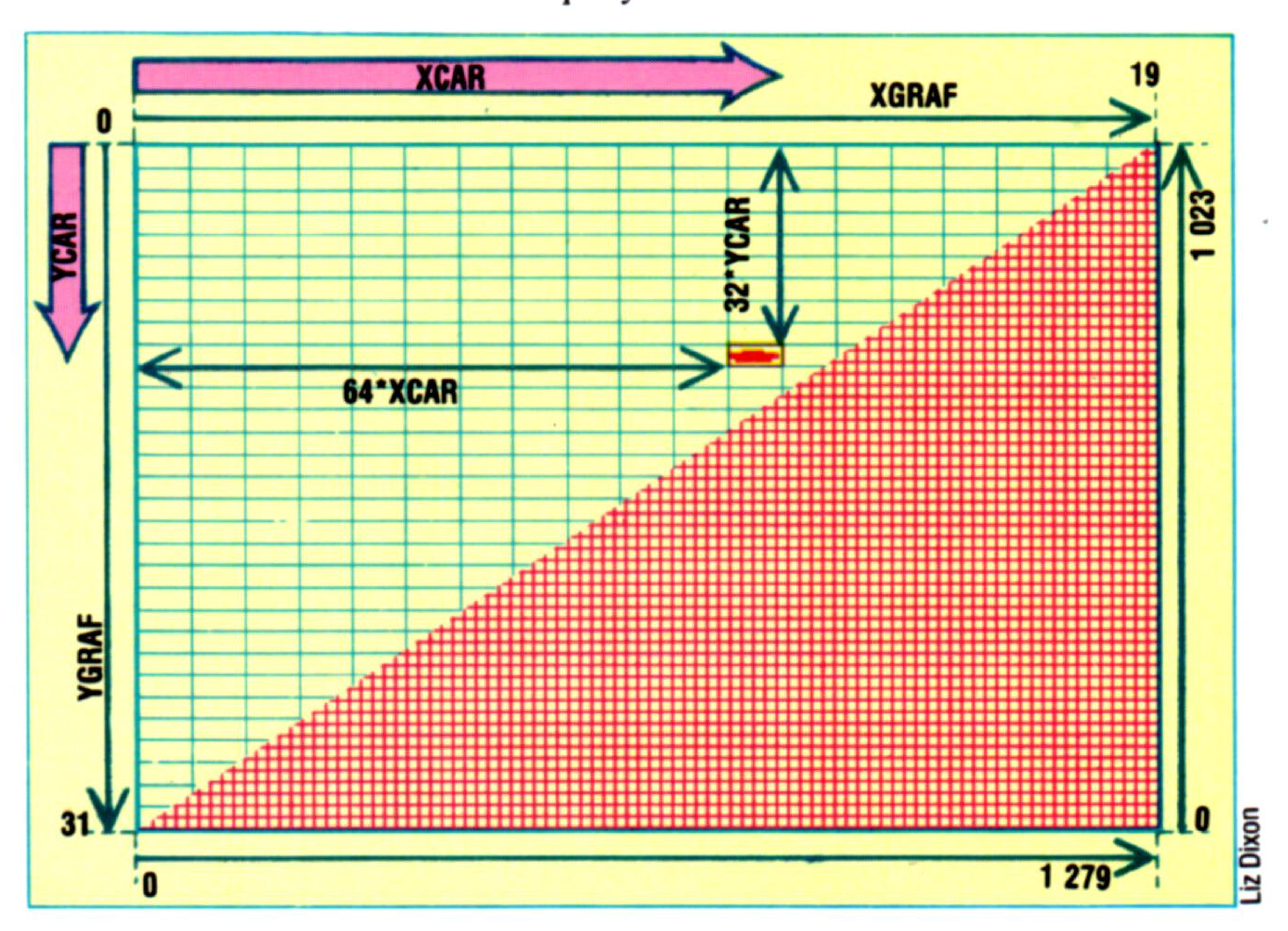
Tal vez se desee alterar esta condición y las otras tres condiciones de márgenes para proporcionar un efecto de rodillo en todos los bordes del campo.

Una de las reglas de nuestro juego es que mientras el jugador desplaza el detector de minas por el campo destruyendo minas, el ayudante del jugador imita todos los movimientos. Para hacer esto, debemos actualizar automáticamente las coordenadas del ayudante, que se relacionan con las coordenadas del detector mediante un sencillo par de fórmulas, que podemos ver en las líneas 3360 y 3370. Para demostrar cómo las mismas producen movimientos de espejo, observemos la relación existente entre las coordenadas x (xhom = 19−xdet).

Inicialmente, xdet es 2 y xhom es 17. Si el detector se mueve un lugar a la derecha, xdet aumenta a 3. Utilizando la fórmula anterior, xhom se calculará como 19−3 = 16. Esto significa que el ayudante se mueve un lugar hacia la izquierda. Si se vuelve a mover xdet hacia la derecha, entonces xdet se convierte en 4 y xhom será 15, y así sucesivamente. Las coordenadas y funcionan de modo similar.

Antes de imprimir (PRINT) el detector y el ayudante nos queda por realizar aún otra tarea. Debemos comprobar si el ayudante o el detector se han movido a una celda de caracteres que ya está ocu-

Mapa
El juego mezcla gráficos en alta resolución con la visualización de textos BBC/Electron. Ello tiene sus ventajas pero significa que se deben mezclar dos sistemas distintos de coordenadas, uno para gráficos y otro para textos. El BBC/Electron posee varios formatos de texto diferentes y, por tanto, cada uno de ellos posee su propio sistema de coordenadas. El juego utiliza la modalidad 5, que proporciona 20 caracteres a lo ancho de la pantalla y 32 a lo alto. Esto se puede apreciar en las partes superior e izquierda del diagrama.
Las máquinas poseen, asimismo, tres resoluciones para gráficos distintas, aunque por suerte todas ellas utilizan el mismo sistema de coordenadas. Éste trata a todas las modalidades como si tuvieran una resolución de 1 280 por 1 024. Esto se puede apreciar en las partes inferior y derecha del diagrama.
El programa utiliza el sistema de coordenadas de alta resolución para leer puntos de la visualización de textos en baja resolución. Ello significa convertir la posición de un carácter a una coordenada de alta resolución. Para hacer esto se debe multiplicar por 64 la coordenada horizontal (XCAR, en el programa) y por 32 la coordenada vertical (YCAR). También se debe superar otro problema. Las coordenadas de la pantalla para texto empiezan desde 0 en la parte superior de la pantalla y se cuentan hacia abajo, mientras que las coordenadas para gráficos empiezan desde 0 en la parte inferior y se cuentan hacia arriba. Esto se resuelve fácilmente restándole 32*YCAR a 1 023



pada por una mina. El BASIC BBC nos permite investigar cualquier punto de la pantalla para ver si está encendido con un color determinado. POINT(X,Y) devolverá el color del pixel de la posición (X,Y). Podemos utilizar esta instrucción para ver si el color de la celda a la que nos estamos moviendo es verde (en cuyo caso contendría una mina). Sólo hay un inconveniente: POINT(X,Y) utiliza el sistema de coordenadas en alta resolución para especificar las coordenadas del punto a examinar. Si deseamos utilizar esta instrucción, debemos antes convertir nuestras coordenadas de celdas de carácter en coordenadas de gráficos.

En el capítulo anterior calculamos que en la modalidad 5 cada celda de caracteres es de 64 unidades de gráficos de anchura por 32 unidades de gráficos de altura (véase p. 884). Multiplicando xcar por 64 obtendremos la coordenada xgraf del margen de la celda en cuestión. Sumándole a xgraf otros 32 obtendremos la coordenada x del centro de la celda. El cálculo de ygraf es algo más complicado porque los dos sistemas trabajan en sentidos opuestos. En la parte superior de la pantalla ygraf es 1023. Yendo hacia abajo, 32*ycar nos llevaría hasta la parte superior de la celda especificada; avanzando otras 16 unidades hacia abajo llegaríamos a la coordenada y del centro de la celda. Así, el siguiente procedimiento se puede diseñar para convertir coordenadas de caracteres en coordenadas de gráficos:

```
3720 DEF PROCconvertir(xcar,ycar)
3730 xgraf = 64*xcar + 32
3740 ygraf = 1023-(32*ycar + 16)
3750 ENDPROC
```

Podremos apreciar el verdadero valor que tiene la posibilidad de pasar parámetros entre procedimientos si volvemos a analizar el procedimiento "mover". El procedimiento "convertir" se utiliza dos veces: en primer lugar, para convertir las coordenadas del ayudante en coordenadas de gráficos, y estos valores se utilizan luego en la línea 3390 para comprobar el color verde. (Recuerde que a pesar de que se han actualizado las coordenadas del ayudante, el carácter todavía no ha sido impreso —PRINT— en su nueva posición.) Si el color es verde, entonces el programa salta a otro procedimiento para visualizar una explosión. El procedimiento "convertir" se utiliza por segunda vez en la línea 3400, pero en esta ocasión se calculan las coordenadas del carácter del detector. La línea 3410 verifica luego si la celda está ocupada por una mina. De ser así, se llama al procedimiento "descubierta-mina". Por último se imprimen (PRINT) el detector y el ayudante en sus nuevas posiciones llamando a "situar-sujetos".

En el próximo capítulo analizaremos el procedimiento "explotar"; por ahora colocaremos un procedimiento ficticio. Entre las siguientes líneas:

```
3550 DEFPROCexplotar(x—explosión,y—explosión)
3560 PRINT "BANG"
3570 END
3580 ENDPROC
```

Por último, examinemos el procedimiento "descubierta-mina", al que se llama cuando el detector se desplaza a una celda que está ocupada por una mina. Sería una buena idea introducir un efecto sonoro que indicara que se ha descubierto una mina. Más adelante en el proyecto analizaremos con más profundidad el sonido, así que de momento todo cuanto necesitamos saber es que la sentencia SOUND de la línea 3790 produce un "ping" de tono agudo. Sin embargo, la principal función de esta rutina es la de incrementar el tanteo del jugador. El programa utiliza dos variables para el tanteo, la primera de las cuales es una variable numérica que se incrementa a razón de 150. Para que el tanteo se imprima siempre como un número de cinco dígitos debemos agregarle ceros por la izquierda. Con el fin de hacer esto, debemos primero convertir el valor numérico del tanteo a una variable string o en serie y utilizar después técnicas de manipulación de series, como las descritas previamente en el proyecto (véase p. 884) para agregar ceros por delante. El procedimiento completo es:

```
3770 DEF PROCdescubierta—mina
3780 REM ** EFECTO SONORO **
3790 SOUND 2,-15,170,3
3800 REM ** INCREMENTAR TANTEO **
3810 COLOUR 2
3820 tanteo = tanteo + 150
3830 tanteo$ = STR$(tanteo)
3840 tanteo$ = LEFT$(cero$,5-LEN(tanteo$)) + tanteo$
3850 PRINTTAB(11,28);tanteo$
3860 ENDPROC
```

En la última parte del proyecto escribimos un corto programa de llamada para los procedimientos que hemos escrito hasta ahora. Los procedimientos reseñados aquí se pueden añadir a su programa con los números de línea consignados. La única modificación necesaria es llamar al procedimiento "leer-teclado" desde dentro del bucle principal del programa de llamada. Por consiguiente, deberá agregar temporalmente a su programa esta línea:

```
55 PROCleer-teclado
```



Ian McKinnell/Liz Heaney

Disparar al objetivo

En este punto del proyecto del Campo de Minas, el programa llenará la pantalla con minas colocadas al azar; creará la imagen suya y una imagen espejo; definirá las tablas del marcador y proporcionará movimiento. Debido a que el programa aún no está completo, si usted cae sobre una mina sólo verá la palabra "BANG" impresa en la pantalla. En el próximo capítulo de este proyecto diseñaremos una rutina que cree una verdadera explosión con los efectos sonoros correspondientes. También es posible que se encuentre con mensajes de error en ciertos puntos de la ejecución del juego. Estos mensajes se producen debido a la estructura incompleta del juego e irán desapareciendo a medida que se vayan añadiendo las subrutinas finales. No obstante, ya hemos llegado a un punto en el cual el programa ha asumido todas las características de un juego de acción rápida

Acción refleja

El programa que presentamos le permite medir sus tiempos de reacción participando en un juego sencillo

A todos nos gusta creer que tenemos muy buenos reflejos; de hecho, tendemos a suponer que nuestras reacciones ante los acontecimientos son casi instantáneas. Sin embargo, es probable que nuestra respuesta más veloz ante un estímulo sea más o menos de un tercio de segundo, que parece suficientemente rápida hasta que uno considera que un coche a gran velocidad habrá recorrido unos 10 metros durante ese lapso. Raramente los tiempos de reacción son constantes; el exceso de alcohol, la fatiga o la enfermedad pueden tener un efecto negativo sobre los reflejos. Nuestro programa está diseñado para probar los reflejos de cualquier persona y, para obtener una mayor exactitud, se calcula un promedio de los tiempos obtenidos en cinco pruebas.

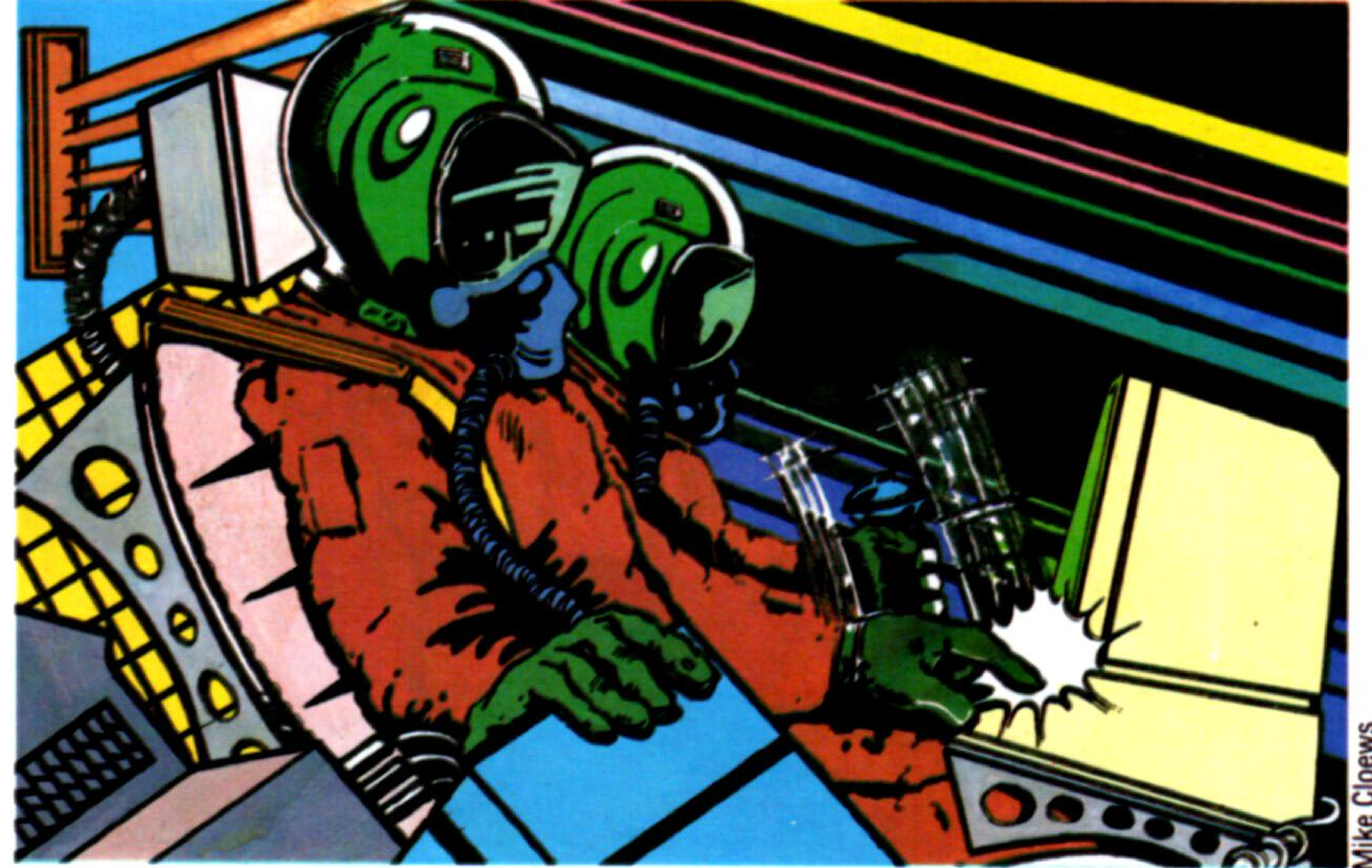
El jugador asume el papel del piloto de una nave espacial que debe entregar con toda urgencia un cargamento de medicinas a una colonia del cinturón de asteroides. La velocidad es vital para que los medicamentos lleguen a tiempo, pero la nave se debe detener tan pronto como exista la inminencia de una colisión. Simplemente pulsando la barra espaciadora la nave se detendrá, y el jugador podrá entonces seguir abriéndose camino a través de los asteroides. El juego visualiza el tiempo que transcurre entre la aparición en pantalla de los asteroides y la pulsación de la barra espaciadora. Después de que se haya detenido la nave en cinco ocasiones, se calcula el tiempo de reacción promedio. Para asegurarse de que el jugador no haga trampas, el programa verifica si la barra espaciadora se está manteniendo pulsada de forma continua; de ser así, los motores se paran y se oye un sonido de advertencia. Al mismo tiempo que aparecen los asteroides en la pantalla, el radar de la nave emite una nota aguda. Como ejercicio, intente cambiar el programa de modo que se reciba un aviso audible o bien visual, pero no ambos a la vez.

Una modificación final podría ser seleccionar entre dos o más opciones presentadas en la pantalla. Ello haría que el programa fuera más una prueba de decisiones y menos una indicación de simples tiempos de reacción.

Tiempo de comprobación

```
10 REM *PARA EL SPECTRUM
20 INK 7: PAPER 0: BORDER 0
30 FOR T = 0 TO 31: READ A
40 POKE USR "A" + T,A: NEXT T
60 DATA 123,231,255,255,245,233,123,100
70 DATA 34,187,208,49,86,96,178,205
80 DATA 67,234,45,123,198,255,29,245
90 DATA 78,100,245,60,50,160,245,189
150 LET M = 0: LET A = 0: LET V = 0: DIM R(5): CLS
220 FOR G = 1 TO 5
250 FOR P = 1 TO RND*150 + 50
260 PLOT INT (RND*250),INT (RND*170)
270 IF INKEY$ <> " " THEN GO TO 280
275 BEEP .1,-3: GO TO 265
280 NEXT P
290 FOR A = 144 TO 147
320 PRINT AT RND*20,RND*30;CHR$(A)
330 NEXT A
350 BEEP .05,15: LET C = 0
360 LET C = C + 1: LET A$ = INKEY$
410 IF A$ <> " " THEN GO TO 355
420 LET R(G) = C/66 + .05
430 PRINT AT 19,0;"HA TARDADO";INT (R(G)*100)/100;
440 PRINT " SEGUNDOS EN PARAR"
445 PRINT "LOS MOTORES"
450 LET M = M + R(G)
453 FOR J = 1 TO 300: NEXT J
455 NEXT G
460 LET A = M/5: CLS
480 PRINT "AL CABO DE CINCO PRUEBAS"
490 PRINT "SU TIEMPO DE REACCION MEDIO FUE";INT (A*100)/100;
495 PRINT " SEGUNDOS"
500 FOR G = 1 TO 5: LET V = V + ABS (R(G)-A)
510 NEXT G
520 PRINT: PRINT "SU TIEMPO DE REACCION VARIO EN UN"
530 PRINT INT (V*20/A);" POR CIENTO"
540 PRINT: PRINT "QUIERE OTRA PASADA?(S/N)"
550 LET R$ = INKEY$: IF R$ = "s" THEN GO TO 150
560 IF R$ <> "n" THEN GO TO 550
570 INK 0: PAPER 7: BORDER 7: CLS

10 REM *PARA EL BBC MICRO Y ELECTRON
20 MODE 1: CLS: VDU 23;8202;0;0;0: DIM REAC (5)
30 REM *PREPARAR CARACTERES (ASTEROIDES)
40 FOR J = 0 TO 3
50 VDU 23,237 + J,78,100,245,60,50,160,245,189
60 VDU 23,223 + J,89,67,34,156,123,200,256,29
70 NEXT J
80 SUMA = 0: PROMEDIO = 0: VARIAC = 0: CLS
90 FOR VEZ = 1 TO 5
100 REM *TRAZAR ESTRELLAS
110 FOR DEMORA = 1 TO 100 + RND(300)
120 PLOT 69,RND(1200),RND(1012)
130 IF INKEY(-99) = 0THEN 150 ELSE
140 SOUND 1,-15,5,1: GOTO 130
150 NEXT DEMORA
160 REM *TRAZAR ASTEROIDES EN PANTALLA
170 FOR ASTEROIDE = 233 TO 240
180 X = RND(28): Y = RND(28)
190 PRINT TAB(X,Y);CHR$(ASTEROIDE)
200 NEXT ASTEROIDE
210 SOUND 2,-15,150,3
220 REM * COMPROBAR BARRA ESPACIADORA
230 TIEMPO = 0
240 IF INKEY(-99) = 0 THEN 240
250 REAC(VEZ) = TIEMPO/100
260 PRINT TAB(3,26);"HA TARDADO ";
270 PRINT;REAC(VEZ);" SEGUNDOS "
280 PRINT TAB(3,28);"EN PARAR LOS MOTORES"
290 FOR I = 1 TO 500: NEXT I
300 SUMA = SUMA + REAC(VEZ)
310 NEXT VEZ
320 FOR I = 1 TO 3000: NEXT I
330 REM *CALCULAR TIEMPO RESPUESTA
340 PROMEDIO = SUMA/5
350 FOR VEZ = 1 TO 5
360 VARIAC = VARIAC + ABS(REAC(VEZ)-PROMEDIO)
370 NEXT VEZ
380 CLS: PRINT: PRINT
390 PRINT "AL CABO DE CINCO PRUEBAS SU TIEMPO"
400 PRINT "DE REACCION MEDIO FUE"; PROMEDIO;" SEGUNDOS"
410 PRINT: PRINT "SU TIEMPO DE REACCION VARIO EN UN"
420 PRINT INT(VARIAC*20/PROMEDIO);" POR CIENTO"
430 PRINT: PRINT "QUIERE OTRA PASADA?S/N"
440 R$ = INKEY$(0): IF R$ = "S" THEN 80
450 IF R$ <> "N" THEN 440 ELSE MODE 4
```


Mike Cloews

Círculo luminoso

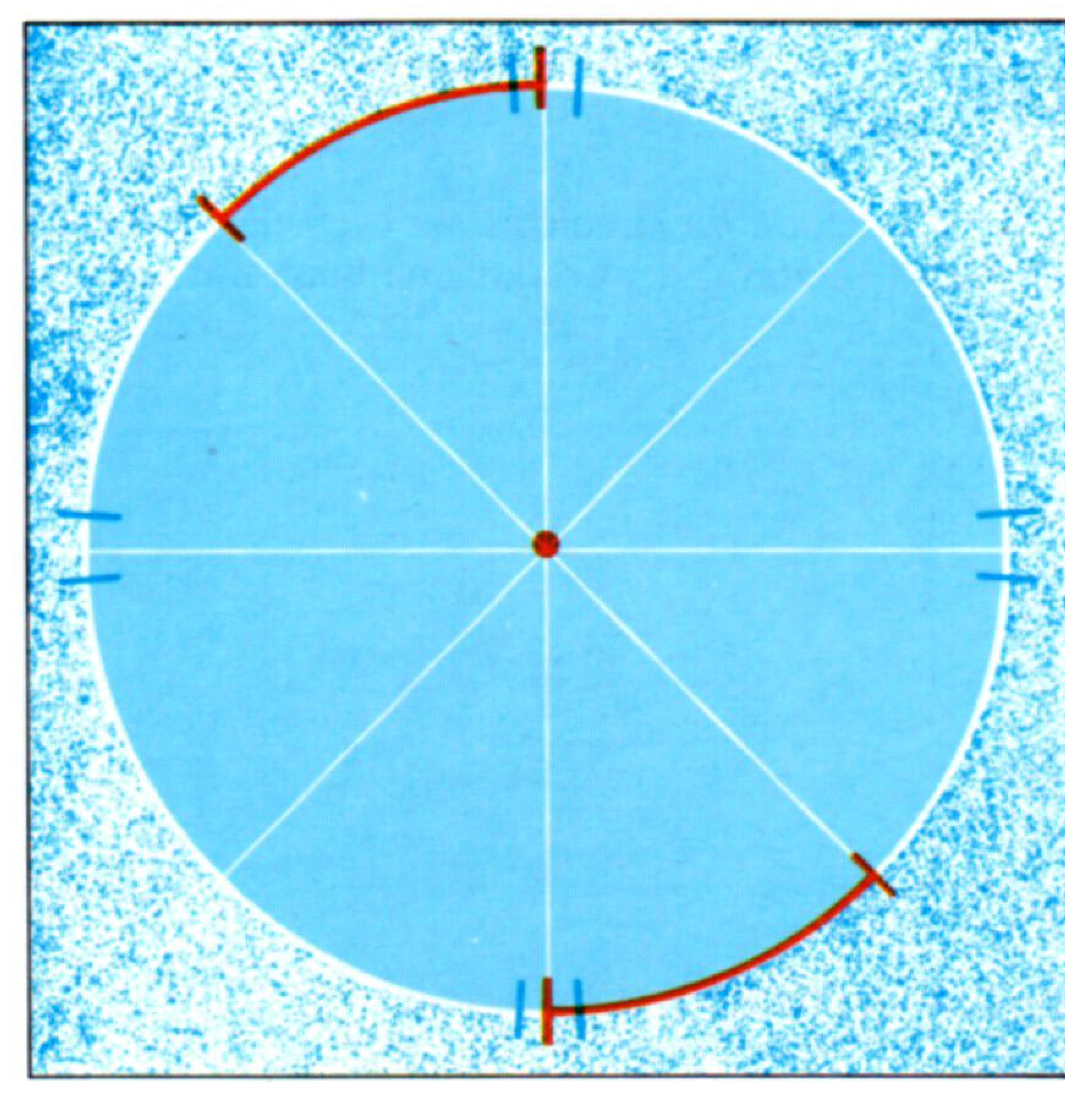
Vamos a comprobar cómo funciona una rutina en lenguaje máquina que sirve para crear círculos y pueda acoplarse con el lenguaje BASIC

Dibujar una circunferencia es una cosa muy sencilla, evidentemente mucho más que dar con la fórmula matemática que permita trazar sus puntos. El modo más sencillo de dibujarla se consigue sirviéndonos de las funciones COS (coseno) y SIN (seno), como en este ejemplo:

```
DEF PROCCIRCLE (XORG,YORG,R)
  MOVER XORG + R,YORG
  FOR ALFA = 0 TO 2*PI STEP PI/32
    X = R*COS(ALFA)
    Y = R*SIN(ALFA)
    TRAZAR X + XORG,Y + YORG
  NEXT ALFA
FIN PROC
```

Pero este PROCedimiento lleva su tiempo de ejecución, porque emplea dos funciones, COS y SIN, cuyos cálculos son lentos. La circunferencia se va dibujando con relativa lentitud. Sin embargo, es posible acelerar su trazado mediante el siguiente algoritmo, derivado de la geometría elemental y del cálculo diferencial:

```
 3 MODO1
 5 PROCCIRCLE (500,600,200)
 7 END
10 DEF PROCCIRCLE (XORG,YORG,R)
20 Y = R
30 FOR X = 1 TO Y*.707
40 Y = Y - X/Y
50 PROCPOINTS(X,Y)
60 NEXT
70 ENDPROC
80 DEF PROCPOINTS (X,Y)
90 PLOT 69,XORG+X,YORG+Y
100 PLOT 69,XORG-X,YORG+Y
110 PLOT 69,XORG-X,YORG-Y
120 PLOT 69,XORG+X,YORG-Y
130 PLOT 69,XORG+Y,YORG+X
140 PLOT 69,XORG-Y,YORG+X
150 PLOT 69,XORG-Y,YORG-X
160 PLOT 69,XORG+Y,YORG-X
170 ENDPROC
```


Liz Dixon

Imágenes especulares
El programa en código máquina utiliza una ecuación seleccionada especialmente por su velocidad para trazar el círculo. No obstante, no hace falta calcular la posición de todos los puntos de éste. En realidad, la mitad superior de la circunferencia es una imagen especular (como reflejada en un espejo) de la mitad inferior, por tanto es suficiente calcular sólo media circunferencia. De igual modo, la mitad derecha es una imagen especular de la mitad izquierda.
En la práctica, el programa únicamente calcula una octava parte del círculo. Cada punto de este octavo es copiado siete veces en otras partes del círculo para llegar a trazarlo en su totalidad

Esta rutina dibuja la circunferencia en ocho zonas a la vez, lo que evidentemente acelera el proceso. Este algoritmo es además mejor que el inicial pues no emplea las funciones de seno y coseno que debían calcularse antes del trazado de cada punto. Sin embargo, todavía resulta algo lenta la ejecución del proceso si se piensa que se debe calcular una división.

Este otro procedimiento alternativo que representamos a continuación no necesita ningún cálculo aritmético complicado:

```
 10 MODO 4
 20 PNUM = 69
 30 PROCCIRCLE(500,600,200)
 40 END
 50 :
 60 DEF PROCCIRCLE(X,Y,R)
 70 VDU29,X;Y;:REM ESTABLECER ORIGEN
    DEL GRAFICO
 80 X = 0:Y = R:D = 3-2*R:REM
    VARIABLES
 90 REPEAT
100 PROCCPLOT
110 IFD < 0:D = D+4*X+6:ELSED =
    D+4*(X-Y)+10:Y = Y-4
120 X = X+4
130 UNTIL X > Y:ENDPROC
140 :
150 DEF PROCCPLOT
160 PLOT PNUM,X,Y
170 PLOT PNUM,Y,X
180 PLOT PNUM,Y,-X
190 PLOT PNUM,-X,Y
200 PLOT PNUM,-X,-Y
210 PLOT PNUM,-Y,-X
220 PLOT PNUM,-Y,X
230 PLOT PNUM,X,-Y
240 ENDPROC
```

El método se conoce como algoritmo de Breshen. Resulta mucho más rápido al no tener que realizar suma, resta o multiplicación alguna por dos o por cuatro (que pueden realizarse mediante desplazamientos de bits). Es éste el algoritmo que usaremos en nuestro programa en lenguaje máquina para dibujar una circunferencia.

Para permitir que el programa funcione en todas



Círculos BBC

```
30PNUM=69
40S%=2
50OSWRCH=&FFEE
60DIM CODE% 600
70DIM D% 12
80X=D%+2
90Y=D%+4
100D=D%+6
110N1=D%+8
120N2=D%+10
130PROCCOMPILE
140MODE4
150PROC_OLYMPIC
160END
170#
180DEF PROCCOMPILE
190FOR I% =0 TO S% STEP S%
200K%=P%
210P%=CODE%
220[OPT I%
230.CIRCLE
240JSR INIT
250:
260.LOOP
270JSR COMPXY:BMI DOIT:JSR CPLOT:RTS
280.DOIT
290JSR CPLOT
300LDA D+1:BPL D_IS_POS
310:
320.D_IS_NEG
330JSR DNEG
340JSR ADD_4_TO_X
350JMP LOOP
360:
370.D_IS_POS
380JSR DPOS
390JSR ADD_4_TO_X
400JMP LOOP
410:
420.INIT
430LDY #8
440.L7
450LDA &601,Y:STA &80,Y
460DEY:BPL L7
470INY
480LDA (&80),Y:STA X
490LDA (&83),Y:STA Y
500LDA (&86),Y:STA D
510INY
520LDA (&80),Y:STA X+1
530LDA (&83),Y:STA Y+1
540LDA (&86),Y:STA D+1
550LDA #29:STA D%+1
560LDA #0:STA D%
570JSR PSTR
580LDA #25:STA D%
590LDA #PNUM:STA D%+1
600JSR SETD
610RTS
620:
630.COMPXY
640LDA X:STA N1:LDA X+1:STA N1+1
650LDA Y:STA N2:LDA Y+1:STA N2+1
660JSR SUB
670LDA N1+1
680RTS
690:
700.CPLOT
710LDX #4
720.L2
730JSR P2
740DEX:BNE L2
750RTS
760:
770.DNEG
780LDA X:STA N1:LDA X+1:STA N1+1
790JSR TIMES4
800LDA #6:STA N2:LDA #0:STA N2+1
810JSR ADD
820LDA D:STA N2:LDA D+1:STA N2+1
830JSR ADD
840LDA N1:STA D:LDA N1+1:STA D+1
850RTS
860:
870.DPOS
880LDA X:STA N1:LDA X+1:STA N1+1
890LDA Y:STA N2:LDA Y+1:STA N2+1
900JSR SUB
910JSR TIMES4
920LDA #10:STA N2:LDA #0:STA N2+1
930JSR ADD
940LDA D:STA N2:LDA D+1:STA N2+1
950JSR ADD
960LDA N1:STA D:LDA N1+1:STA D+1
970JSR SUB_4_FROM_Y
980RTS
990:
1000.ADD_4_TO_X
1010LDA #4:STA N1:LDA #0:STA N1+1
1020LDA X:STA N2:LDA X+1:STA N2+1
1030JSR ADD
1040LDA N1:STA X:LDA N1+1:STA X+1
1050RTS
1060:
1070.SUB_4_FROM_Y
1080LDA#4:STA N2:LDA #0:STA N2+1
1090LDA Y:STA N1:LDA Y+1:STA N1+1
1100JSR SUB
1110LDA N1:STA Y:LDA N1+1:STA Y+1
1120RTS
1130.SETD
1140LDA #0:STA X:STA X+1
1150LDA D:STA Y:LDA D+1:STA Y+1
1160ASL D:ROL D+1
1170LDA #3:STA N1:LDA #0:STA N1+1
1180LDA D:STA N2:LDA D+1:STA N2+1
1190JSR SUB
1200LDA N1:STA D:LDA N1+1:STA D+1
1210RTS
1220:
1230.P2
1240JSR PSTR
1250JSR SWAPXY
1260JSR PSTR
1270JSR NEGY
1280RTS
1290:
1300.TIMES4
1310ASL N1:ROL N1+1
1320ASL N1:ROL N1+1
1330RTS
1340:
1350.ADD
1360CLC
1370LDA N1:ADC N2:STA N1
1380LDA N1+1:ADC N2+1:STA N1+1
1390RTS
1400:
1410.SUB:\ (N1=N1-N2)
1420SEC
1430LDA N1:SBC N2:STA N1
1440LDA N1+1:SBC N2+1:STA N1+1
1450RTS
1460:
1470.PSTR
1480LDY #250
1490.L1
1500LDA D%-250,Y
1510JSR OSWRCH
1520INY
1530BNE L1
1540RTS
1550:
1560.SWAPXY
1570LDA X:PHA:LDA X+1:PHA
1580LDA Y:STA X:LDA Y+1:STA X+1
1590PLA:STA Y+1:PLA:STA Y
1600RTS
1610:
1620.NEGY
1630LDA #0:STA N1:STA N1+1
1640LDA Y:STA N2
1650LDA Y+1:STA N2+1
1660JSR SUB
1670LDA N1:STA Y
1680LDA N1+1:STA Y+1
1690RTS
1700]
1710NEXT
1720ENDPROC
1730#
1740DEF PROCCIRCLE(P1%,P2%,P3%)
1750CALL CIRCLE,P1%,P2%,P3%
1760ENDPROC
1770#
1780DEF PROC_OLYMPIC
1790PROCCIRCLE(300,600,150)
1800PROCCIRCLE(650,600,150)
1810PROCCIRCLE(1000,600,150)
1820PROCCIRCLE(475,450,150)
1830PROCCIRCLE(825,450,150)
1840VDU29,0;0;
1850MOVE100,250
1860DRAW100,800
1870DRAW1200,800
1880DRAW1200,250
1890DRAW100,250
1900ENDPROC
```

Esta rutina facilita el dibujo de circunferencias en el BBC y en el Electron. Para utilizar la rutina en código máquina basta con colocar los valores adecuados en tres variables INTEGER (p. ej., X%,Y% y R%) y llamarla con la instrucción: CALL CIRCLE,X%,Y%,R%. Dibujará una circunferencia de radio R% y centro en X%,Y%. Obsérvese que el origen del gráfico es desplazado hacia el centro de la circunferencia por la rutina. Pero lo podemos restablecer con VDU29,0;0;. Por desgracia, la instrucción CALL no posibilita utilizar las expresiones como parámetros, por lo que X%,Y% y R% sólo pueden ser variables. Para obviar este inconveniente nos hemos servido del procedimiento PROCCIRCLE (como se ve en el listado)

las modalidades de gráficos, hemos empleado la instrucción VDU 25 para trazar todos los puntos de la circunferencia, a conciencia de que esto retrasa considerablemente la ejecución. El programa está estructurado en subrutinas autosuficientes de modo que ayude a su comprensión. Y para evitar confusiones se han utilizado sólo técnicas directas. El precio en la claridad de comprensión ha sido la lentitud en la ejecución del programa. Así y todo, para el trazado de una circunferencia de 300 unidades de radio se necesitan 0,52 segundos en código máquina frente a los 1,9 segundos en nuestra versión original en BASIC.

Pueden hacerse útiles mejoras a la rutina alterando el valor de PNUM en la línea 30, cambiando del 69 al 5. Esto hará que el programa dibuje líneas en lugar de puntos por lo que creará un disco sólido de color en vez de un círculo. Este cambio provocará el efecto colateral de dibujar una línea no deseada por cada círculo. Para evitarla, hay que añadir esta sentencia:

```
1745 VDU29,0;0;:MOVE P1%,P2%n
```

Lo mismo se puede hacer en assembler con la siguiente línea:

```
575 LDA#25:JSR OSWRCH:LDA#4:JSR
    OSWRCH:LDA#0:JSR OSWRCH:LDA#0:JSR
    OSWRCH:LDA#0:JSR OSWRCH:LDA#0:JSR
    OSWRCH
```

Realizando mejoras más complicadas, el programa llegará a dibujar arcos y elipses.

Estrella solitaria

A la firma norteamericana Texas Instruments se la puede considerar la creadora de la revolución del ordenador personal



El presidente de TI
El presidente y principal ejecutivo de Texas Instruments, J. Fred Bucy, ocupa el cargo de presidente desde abril de 1976

Introducido en 1978, el ordenador personal TI99/4A utilizaba el microprocesador de 16 bits TMS9900, de Texas Instruments, y tenía 16 K de memoria RAM. La máquina disponía de una selección de 16 colores y tres canales de sonido. A pesar de las excelentes ventas, el ordenador, sin embargo, fue víctima de la implacable guerra de precios que tuvo lugar en Estados Unidos en 1982 y 1983. A consecuencia de ella, a principios de este último año TI abandonó por completo la producción de la máquina.

Richard Mann, director de relaciones públicas para Europa de Texas Instruments, afirma que "el TI99/4A se estaba vendiendo muy bien, pero no era rentable y perdimos varios centenares de millones de dólares". A la pregunta de si TI tiene previsto lanzar otra máquina, responde: "Rotundamente, no. Eso lo dejamos bien claro en su momento. A nosotros nos interesa más vender máquinas de cientos de dólares que máquinas de decenas de dólares". No obstante, TI tiene la intención de mantenerse "firmemente en el mercado del ordenador profesional".

Los otros productos de la empresa van desde juguetes electrónicos para niños, como el *Speak and Spell*, a miniordenadores y sofisticados equipos para detección de movimientos sísmicos. Posee 15 plantas de producción distribuidas por todo el mundo y un movimiento anual de cuatro billones de dólares.

Texas Instruments fue fundada en 1930 por dos científicos, John Karchner y Eugen McDermott, y originalmente se denominó Geophysical Service Incorporated. Karchner había estado investigando sobre la idea de hacer rebotar ondas sonoras en los estratos geológicos para calcular su profundidad, y la empresa se estableció en la ciudad de Dallas (Texas), para venderle esta idea a la industria petrolífera.

Geophysical Service Incorporated fue creciendo constantemente durante los años treinta, desarrollando nuevas técnicas y equipos para peritación sismológica, y durante la segunda guerra mundial sus equipos de peritación resultaron de gran utilidad para detectar submarinos, lo que determinó que GSI creara un departamento de laboratorio y fabricación. En 1951 esta rama de la empresa matriz había crecido hasta tal punto que se decidió establecerla por sí sola. La nueva empresa fue bautizada como Texas Instruments.

Al año siguiente, TI obtuvo una licencia de Bell Laboratories para fabricar los transistores, que se habían inventado recientemente, y en 1954 produjo la primera radio a transistores del mundo. En 1958 Jack Kilby, un ingeniero de la empresa, inventó el circuito integrado, y desde entonces TI ha permanecido en la vanguardia de la investigación en este campo. Kilby también colaboró en la invención de la primera calculadora electrónica de mano del mundo, en 1967.

El ordenador TI Professional, que fue lanzado al mercado en enero de 1983, está basado en el procesador 8088 y puede, por consiguiente, ejecutar CPM-86 y MS-DOS. Posteriormente, durante ese mismo año, la empresa introdujo el ordenador de oficina portátil Texas Instruments y el micro de mano CC 40.

Actualmente, la empresa también produce una amplia variedad de componentes electrónicos, incluyendo chips de RAM de 64 Kbytes, de los que declara ser uno de los mayores proveedores de todo el mundo. Los chips de TI aparecen en casi todos los ordenadores personales, incluyendo el Commodore 64 y el Sinclair Spectrum. Al igual que la mayoría de los otros fabricantes de chips, TI ha desarrollado una gama de procesadores de 16 bits, entre la que se incluye el TMS9900, incorporado en el TI99/4A.

Este procesador fue inusual por el hecho de no tener registros internos, que se incluyeron en una "memoria temporal de trabajo" (*sratchpad memory*) especial exterior a la CPU. Richard Mann afirma: "El TMS 9900 estuvo un poco adelantado para su tiempo. Tenía muchas características eficaces, pero no logramos que se aceptara su arquitectura". Ahora Texas Instruments ha comercializado un procesador de 16 bits denominado TMS99000 y la empresa confía en que éste obtendrá un mayor éxito entre los usuarios.



Planta de Texas Instruments
Gran parte de la fabricación de los equipos de Texas Instruments se realiza en estas modernas instalaciones Expressway en Dallas (Texas)



Editorial Delta, S.A.



10046
9 788485 822836